SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij

PRILAGODBA KORISNIČKOG SUČELJA MOBILNOG UREĐAJA EMOCIJAMA ZASNOVANA NA ANALIZI TEKSTA

Diplomski rad

Karlo Boni

Osijek, 2016.

SADRŽAJ

1. UVOD

U posljednjih nekoliko desetljeća sve je više istraživanja na području psihologije koja se bave ispitivanjem kako razni čimbenici, kao na primjer boje, slike i drugo utječu na čovjekovo emocionalno stanje. Međutim, opće poznata činjenica je da je čovjek jedan od najsloženijih bića na Zemlji i svaki čovjek je jedinstven, što onemogućuje razvoj potpuno univerzalnog sustava za utjecaj na ljudske emocije. Unatoč tome, razvijeno je nekoliko grana psihologije koje se bave ovom problematikom, kao na primjer psihologija boja, pridonoseći time boljem razumijevanju ljudskih emocija te utjecaja na njih. Ovakve metode se koriste već desetljećima, najviše u marketingu, pri izradi logotipa i reklama.

Cilj aplikacije za Android operacijski sustav, nazvan „Emodapter“, je prepoznavanje emocije korisnika iz napisanog teksta pri dopisivanju s drugim ljudima te na osnovi prepoznate emocije prilagodba grafičkog sučelja mobilnog uređaja. Drugim riječima, koristeći razne metode za utjecaj na ljudske emocije, ovom aplikacijom pokušava se poboljšati trenutno emotivno stanje korisnika. Za postizanje tog cilja koristit će se hibridna metoda prepoznavanja emocija iz napisanog teksta, a na korisnikove emocije će se utjecati mijenjanjem boja tipkovnice, pozadinskom slikom te inspirativnim citatima. Testiranje će se ostvariti prikupljanjem podataka iz aplikacije, a testirat će se provjera točnosti prepoznavanja emocije. Također, testirat će se učinkovitost aplikacije ispitivanjem korisnika nakon korištenja aplikacije.

U drugom poglavlju rada se nalaze statistički rezultati vezani uz razvoj društvenih mreža. U trećem poglavlju je opis osnovnih teorija emocija te utjecaja boja, slika i citata na emocije. U četvrtom poglavlju opisuju se načini automatskog prepoznavanja emocija iz napisanog teksta te njihova problematika i ograničenja. U petom poglavlju navode se korištene tehnologije, a u 6. poglavlju način izrade te korištenje aplikacije. U sedmom poglavlju prikazani su postupci i rezultati testiranja aplikacije „Emodapter“.

2. DRUŠTVENE MREŽE

Rastuća sveprisutnost mobilnih telefona, posebice pametnih telefona, društveno je umrežavanje svela na samo jedan dodir prstom. Prema [1], 40% vlasnika mobitela koristi društvene mreže na svom mobitelu, a 28% njih ih koristi svakodnevno. Može se reći kako se društveno umrežavanje gotovo u potpunosti preselilo na mobilne uređaje. Prema [2], uporaba mobilnih telefona, najčešće pametnih telefona, za pristupanje društvenim mrežama znatno je porasla u posljednjih nekoliko godina. U korporativnom marketingu više nije bilo od značaja kojoj društvenoj mreži korisnici pristupaju, već i s kakvog uređaja joj pristupaju. Rezultati su pokazali kako su pametni telefoni postali u potpunosti dominantni u ulozi uređaja putem kojeg korisnici u pokretu pristupaju društvenim mrežama. Prema istraživanju iz [2], vidljivo je kako korištenje društvenih mreža putem računala i prijenosnih računala tijekom posljednjeg desetljeća opada sa 63%, za 2012. godinu, na 59%, za 2014. godinu. S druge strane, korištenje mobitela i tableta za pristup društvenim mrežama bilježi porast. Prema [2], trenutno 44% odraslih korisnika Interneta pristupaju društvenim mrežama putem mobitela, a nešto više od jedne petine im pristupa putem tableta. Navedeni podaci pokazuju kako komunikacija preko mobilnih uređaja i društvenih mreža postaje sve popularnija. Upravo su ovi čimbenici temelj ovoga rada, odnosno praćenje komunikacije korisnika mobilnog uređaja preko društvenih mreža.

2.1. Značaj društvenih mreža

Prema [4], društvene mreže postale su globalni komunikacijski fenomen. Iako zbog velike izloženosti društvenim mrežama, još uvijek dolazi do zabune koje se mreže mogu smatrati društvenima u velikom broju mrežnih stranica koje su danas dostupne na Internetu. Ono što se može ustanoviti je da društvene mreže postoje već veliki niz godina na internetskom komunikacijskom prostoru i da okupljaju ogroman broj ljudi u komunikaciji na određenim interesnim temama. Društvene mreže su najpopularniji globalni komunikacijski fenomen, jer je čovjek društveno biće te ga stoga društvena komunikacija privlači i zanima. Na društvene mreže ljudi dolaze kako bi se povezali i sprijateljili s novim ljudima, kako bi podijelili odnosno primili ili razmijenili razne informacije. Glavno obilježje društvenih mreža su univerzalna komunikacijska sredstva i tehnike, kao i zajednički interes koji drži grupe ljudi zajedno i samim time daje popularnost društvenoj mreži.

Prema istraživanju iz [5], iz 2009. godine, 59% odraslih Internet korisnika u SAD-u koristi najmanje jednu društvenu mrežu. Usporedbe radi, prema istraživanju [1], iz 2014. godine, taj postotak iznosi

79%. Prema [6], dostupna tehnologija mijenja učenje komuniciranja. Društvene su mreže danas nesumnjivo postale glavni dio dnevnih interakcija s kolegama, prijateljima, ali i drugima s kojima dijelimo zajedničke stavove ili mišljenja. Digitalna komunikacija kroz društvene mreže (i mobilne komunikacije) povezuje osobe 24 sata dnevno svih sedam dana u tjednu, što je bitna razlika u odnosu na tzv. *face-to-face* komunikaciju. U prilog neprekidne dostupnosti društvenih mreža govori istraživanje [7] provedeno među tinejdžerima, kao najbrojnijim korisnica društvenih mreža. Prema njemu, 22% mladih posjećuje društvene mreže više od 10 puta dnevno, dok ih 51% posjećuje više od jednom dnevno. Čak ih 28% na društvenim mrežama dijeli osobne informacije koje inače ne bi dijelili u javnosti.

Prema [4], budućnost društvenih mrežnih stranica leži u osluškivanju potreba korisnika i brzoj prilagodbi na veliku količinu korisnika. Web i društvene mreže postaju potreba u svakom gledištu života i to treba pratiti. Mnogi su shvatili da je princip društvenih mreža koristan i u poslu i u privatnom životu pa su tako počele nastajati specijalizirane društvene mrežne stranice. Sve veći broj korisnika se organizira u razne društvene mreže, koje su namijenjene nekom određenom interesu, jer na taj način se mogu okupiti korisnici sa zajedničkim interesima na jednom mjestu i razmjenjivati sadržaje s drugim korisnicima koji dijele sličan interes, a ne miješaju se s drugim sadržajima i vrstama interesa. Društvene mrežne stranice su postale važan dio svakog društva i njihov utjecaj se ne smije zanemariti, već treba poticati njihov razvoj i rast, ali uz kontrolu privatnosti i poštivanja pravila ponašanja. Iz tih razloga, osluškuju se samo javne reakcije korisnika na određenu temu, ali one ne moraju biti iste svaki puta za istu temu od istog korisnika, već reakcije ovise o trenutnom emotivnom stanju pojedinca. Ova problematika je detaljnije opisana u poglavlju 3.1.

3. EMOCIJE

3.1. Pojam emocija

Emocije su, prema [9], jedan od najvažnijih čimbenika ljudskog života budući da utječu na cjelokupni čovjekov sustav, ali i na međusobni odnos između ljudi. Na svaki događaj ili situaciju čovjek reagira nizom reakcija za koje su one zaslužne. Često zbog njih, na iste događaje ili situacije čovjek može reagirati različitim reakcijama. Emocije sadrže odnos pojedinca s nekim događajem ili objektom, prepoznatljivo mentalno stanje i emocionalno izražavanje te su kratkotrajne, intenzivne uz različite fiziološke promjene i dovode do trenutne promjene ponašanja. Kontroliraju sveukupno ponašanje te utječe na prilagodbu pojedinca.

Prema [9], emocije se ukratko mogu definirati kao skupovi kemijskih ili neuralnih reakcija, koji imaju neku regulacijsku ulogu i na ovaj ili onaj način vode do stvaranja okolnosti povoljnih za organizam. One su biološki određeni procesi koji ovise o urođenom skupu moždanih sklopova, stvorenih tijekom duge evolucijske povijesti. Prema [10], emocije se mogu podijeliti u dvije osnovne skupine: osnovne (jednostavne) i složene (kompozitne). Jednostavne emocije se pojavljuju već kod djece. One su ujedno i univerzalne, jer se pojavljuju u svim kulturama. Njihov je zadatak pripremiti naš organizam za suočavanje s osnovnim problemom preživljavanja. U većini literatura kao osnovne se navode radost, tuga, ljutnja, strah, prihvaćanje, odbijanje (gađenje), očekivanje i iznenađenje. Složene emocije razvijaju se nakon dvije godine starosti. Nastaju miješanjem s jednostavnim emocijama. One nisu od vitalne važnosti, već se radi o društvenim emocijama koje uglavnom ovise o društvenim i kulturalnim utjecajima.

3.2. Teorije o emocijama

Mnogo je pokušaja razumijevanja i definiranja emocija kroz koje se može uočiti veliki broj različitih pristupa. Prema [11], Charles Darwin je, vjerojatno, bio prvi koji je sustavno identificirao i kategorizirao sveobuhvatni raspon emocija. On je smatrao kako emocije predstavljaju mehanizme prilagođavanja i preživljavanja pojedinca. Prema [12], evolucijskoj se teoriji zamjera nedostatak povijesnih dokaza po kojima je nova značajka, zbog povećane prilagodljivosti, zamijenila staru. To se posebice odnosi na psihološke osobine, budući da ne postoje fosili na kojima će se one ispitati. Stoga, teško je dokazati mogućnost prilagodbe emocija. Bez obzira, Darwinova je evolucijska

funkcija emocija priznata od strane suvremenih autora na području emocija. Atraktivnost tog pristupa očituje se u tome što svi ljudi imaju emocije, a isto tako i životinje imaju ponašanje nalik emocijama, što dovodi do zaključka kako su emocije bile prisutne i kod zajedničkog pretka. Prema [11], druga perspektiva je promatranje emocija u biheviorističkom smislu. Gledajući na emocije kao na stanje izazvano podražajem vršenja, propusta ili prestanka nagrađivanja ili kažnjavanja. Ovaj se pristup temelji na jednostavnom razlikovanju zadovoljstva i boli, koja dovodi do različitih mogućih emocija, koje predstavljaju traženje ili izbjegavanje pojedinca za odgovarajućim pozitivnim i negativnim stimulacijama. U ovoj teoriji, emocije su prikazane kao instrument, tj. odgovor pojedinaca na određene podražaje.

Emocije, također, mogu podrazumijevati određene kombinacije (s različitim naglascima i ishodima) fizioloških, psiholoških i psihomotornih komponenta. William James je prema [9], rani zagovornik ovog općeg pristupa, koji je definirao emocije u smislu osjećaja kao „tjelesne ekspresije“, koje slijede percepciju „uzbudljive stvari“. Ostale varijacije poznaju „afektivne“ i „somatske“ dimenzije emocija, „empirijske, bihevioralne i psihologijske“ aspekte ili „tjelesne“ i „kognitivne“ dimenzije emocija. Svi su navedeni pristupi od emocija napravili značajan individualni fenomen. Emocije, međutim, možemo promatrati i u društvenom kontekstu, koji podrazumijeva interakciju sa samim sobom i s drugima. Prema [12], mnogo je teorija razvijeno iz društvene perspektive, ali najznačajnija je ona Jamesa Averilla. Po Averillu „emocija je prolazna društvena uloga (društveno konstituirani sindrom), koja podrazumijeva sposobnost pojedinca za procjenom situacije koja se tumači kao strast, a ne kao djelovanje“. Te prijelazne društvene uloge i sindrome postavljaju društvene norme i očekivanja, iz čega proizlazi kako društvene norme i očekivanja upravljaju emocijama pojedinca.

Najnovija istraživanja vezana uz emocije odvijaju se na području neurobiologije, gdje umjesto da se emocije shvaćaju kao psihološko stanje ili društveni fenomen, na njih se gleda u odnosu na funkciju mozga. Klinička su istraživanja utvrdila kako su emocije povezane sa složenim biološkim procesima u kojem igraju ulogu neurološki, biokemijski i sociokulturalni čimbenici. Darwin je identificirao preko trideset različitih emocija i kategorizirao ih u sedam skupina, grupirajući slične emocije zajedno. James poznaje „grublje emocije“ (tugu, strah, bijes i ljubav) i „suptilnije emocije“ čiji je odjek manje očit i jak. Damasio razlikuje „primarne“ (sreća, tuga, ljutnja, strah i gađenje) i „sekundarne“ emocije. Navedene konstrukcije emocija samo ukazuju na kompleksnost tog fenomena.

3.3. Ekmanova raspodjela emocija

Prema [13], dr. Paul Ekman je jedan od najistaknutijih psihologa te je prema časopisu „Times“ 2009. godine uvršten među 100 najutjecajnijih ljudi 21. stoljeća u svijetu. Njegova istraživanja daju najsnažnije dokaze Darwinovih teorija emocija, odnosno da su izrazi lica, uvjetovani emocijama, urođeni i univerzalni. Među najznačajnijim radovima su prvi i jedini opsežni sustav za objektivno mjerenje pokreta lica, početak korištenja neuralnih mreža za prepoznavanje i mjerenje pokreta lica i dr.

Počeo je proučavati prevare počevši od kliničkih slučajeva u kojima je pacijent tvrdio kako ne osjeća depresiju, a kad je ostao bez nadzora pokušao je učiniti samoubojstvo. Pregledavanjem usporenih snimaka razgovora s pacijentima, Ekman je uočio tzv. mikro izraze lica (engl. *micro facial expressions*), koje su otkrivale jake negativne emocije koje su pacijenti htjeli sakriti. U početnom istraživanju Ekman je svojim ispitanicima pokazivao slike lica osoba s različitim izrazima te su oni trebali za pojedini izraz odrediti emociju. Rezultat istraživanja je bila podjela na šest osnovnih, univerzalnih emocija prepoznatljive prema čovjekovom izrazu lica: sreća, iznenađenje, tuga, ljutnja, gađenje i strah. Na ovoj raspodjeli se temelji većina današnjih sustava za prepoznavanje emocija iz izraza lica. Zahvaljujući velikoj popularnosti Ekmanove raspodjele emocija, one se koriste i u većini drugih sustava za prepoznavanje emocija iz drugih izvora pa tako i iz napisanog teksta.

3.4. Boje

Prema [3], boja je svjetlo, koje putuje od Sunca do nas u obliku valova, na istom elektromagnetskom spektru kao i radiovalovi, mikrovalovi, x-zrake itd. Svjetlost je jedini dio spektra kojeg možemo vidjeti, što objašnjava zašto ju ne shvaćamo toliko ozbiljno kao ostali, nevidljivi dio spektra. Sir Issac Newton je demonstrirao kako svjetlost putuje kao val, kada je pustio bijeli snop svjetlosti kroz prizmu te su se različite svjetlosti pod različitim kutovima razdvojile, tako je dokazao da su boje duge sastavnice bijele svjetlosti. Prema [14], kada svjetlost padne na neki objekt, on će apsorbirati samo one valne dužine koje se poklapaju sa svojom atomskom strukturom, a ostale reflektirati - upravo njih vidimo. Kada svjetlost padne na ljudsko oko, različite valne duljine različito utječu na našu percepciju. U mrežnici one se pretvaraju u električne impulse, koje se šalju u hipotalamus - dio mozga koji upravlja hormonima i našim endokrinim sustavom. Hipotalamus upravlja regulacijom vode,

spavanjem i obrascima ponašanja, uravnoteženosti živčanog sustava, seksualnim i reproduktivnim funkcijama, metabolizmom, apetitom, tjelesnom temperaturom.

Samo je nekoliko naziva za boje, a doslovno ih ima milijun. Stoga je uobičajeno da se riječi za boje, koje nemaju naziv u vlastitom jeziku, posuđuju iz drugih jezika ili im se daju nazivi s obzirom na nijansu i ton. Prema [14], boje su snažan sustav signalizacije koji odašilje priroda. Znanstveno je dokazano kako je ona prva stvar koju zamijetimo kada prilazimo nečemu. Znanost je oduvijek poznavala vezu između boje i raspoloženja, tj. ponašanja te mnogo je istraživanja na tu temu. Uobičajeno je da psiholozi tijekom istraživanja boje definiraju kao „plavu i narančastu“ ili „crvenu i zelenu“ bez ulaženja u detalje oko nijanse i tona. Međutim, većina se znanstvenika slaže da je ljudsko prepoznavanje boja subjektivno te je upravo zato i nepredvidivo.

Važno je napomenuti kako postoji velika razlika između psihologije boja i simbolike boja. Mnogo je primjera simbolike boja. Primjerice, ljubičasta se povezuje s plemstvom, jer je do nedavno bila iznimno skupa te dostupna samo plemstvu, dok je crvena boja krvi i povezuje se s ratom. Ove se asocijacije često poklapaju s psihologijom boja (crvena je često povezana s agresivnošću), ali one nipošto nisu isto. Ključ uspjeha psihologije boja je grupiranje boja po nijansama, u odnosu na određene tipove osobnosti. Svih milijun nijansi i tonova mogu se svrstati u samo četiri skupine nijansi: vedre boje - nježno tople koje ne sadrže crnu, hladne - sadrže više sive, također nježne ali ne nužno svijetle, tople, ali mnogo intenzivnije i vatrene te vrlo jasne i jake boje, bez suptilnosti.

3.4.1. Psihološka svojstva boja

Ljudi se često pitaju vidimo li svi mi boje isto. Prema [14], značajno je kako psihologiji boja nije važno koju boju pojedinac misli da gleda, već utjecaj te boje na njega uzrokovan njezinom energijom. Jedanaest osnovnih boja ima temeljne psihologijske osobine, koje su univerzalne bez obzira na nijansu i ton. Svaki od njih ima potencijalni pozitivni i negativni utjecaj, a koji će prevladati ovisi o kombinaciji boja. One su povezane s tijelom, umom i emocijama te ravnotežom među njima. Prema [14], psihološke osobine jedanaest osnovnih boja prikazane su u tablici u P.3.1.

3.4.2. Utjecaj boje na pojedinca

Autori su u [15] razvili opći model boja i psihološkog funkcioniranja. Po njima je svaka boja nositelj određenog značenja. Boja nije samo stvar estetike, već i prenositelj određene informacije. Također, značenje boja temelji se na dva osnovna izvora: naučenim asocijacijama razvijenim ponavljanim

sparivanjem boja s određenom porukom, idejom ili iskustvom i biološkom tendencijom reakcije na određene boje na određeni način u određenim situacijama. Određene asocijacije na boje proizlaze iz samostalnog učenja, ali teoretičari boja sumnjaju kako je mnogo asocijacija nastalo iz evolucijski ukorijenjene reakcije na određene podražaje. Istraživanja pokazuju kako boje služe kao signali životinjama (npr. crveno voće šalje signal kako je zrelo za jelo), što olakšava prilagodbu ponašanja. Pretpostavlja se kako su i ljudi spremni slično reagirati na podražaj koji šalje boja.

Prema [14], „utjecaj boja“ logičan je i metodičan pristup u okviru psihologije boje. To je jasan i znanstveno razvijeni pristup koji se prakticira još od ranih 80-tih. Kao tvorac ovog pristupa navodi se psihologinja Angela Wright. Pristup se često koristi u komercijalnom dizajnu, odnosno u dizajnu interijera, proizvoda, uniformi i pakiranja, web dizajnu, stvaranju korporativnog identiteta i *brendingu*. Ovaj pristup iza sebe ima gotovo dvadeset uspješnih godina primjene u velikim tvrtkama (*Shell*, *Motorola*, *Procter & Gamble*, *BT and The Body Shop* i dr.), što potvrđuju njihova uspješnost u prodaji vlastitih proizvoda i smanjenje troškova i vremena na dizajn. Boja u mnogim slučajevima može biti sredstvo za prenošenje određene poruke. Razlog tome je što je boja u ulozi komunikatora puno više objektivnija nego što mislimo. Uobičajena je zabluda kako je psihologija čisto subjektivna bez objektivnog kriterija za predviđanje odgovora, vjerojatno, zato što svatko reagira instinktivno i svaki pojedinac ima svoju vlastitu najdražu boju.

3.5. Utjecaj slike na pojedinca

Nedavna istraživanja na području percepcije i spoznaje ukazala su na značaj vizualne percepcije. Prema [16], rezultati tih istraživanju kazuju kako su oči sofisticirana osjetila i glavni ljudski izvor informacija o svijetu. One šalju više podataka u ljudski živčani sustav od bilo kojeg drugog osjetila. Ljudski mozak obrađuje slike na način koji izaziva emocije i instinktivno interpretira stvarnost. U istom trenutku kada se suoči sa slikom, ljudski je mozak naučen interpretirati tu sliku kao stvarnost i trenutno reagirati instinktivnim emocijama.

J.J. Gibson je u svojoj teoriji u [16] objasnio način na koji se interpretira slike. On objašnjava koncept „vidnog polja“ te predstavlja proces svjetlosnog odraza, koji dolazi u naše oči i stvara „vizualni svijet“. U „vizualnom svijetu“ slike interpretiramo kao našu percepciju stvarnosti bez daljnjeg analiziranja konteksta. To podrazumijeva kretanje iz vidnog polja u vizualni svijet bez ulaženja u stadij analitičke obrade. Nadalje, neuroznanstvenik Joseph LeDoux u svojoj studiji [16] navodi kako na gledanje

nečega, prije nego što i stignemo promisliti o tome, prvo odgovaramo emotivno. Ova studija pokazuje kako proces u mozgu djeluje na način da signali koji dolaze do ljudskog oka putuju do talamusa i amigdale (dijela mozga koji igra značajnu ulogu u emotivnim odgovorima) prije no što je drugi signal i poslan u neokortekst. Pojednostavljeno, ljudski mozak radi na način da uzrokuje emotivnu reakciju prije nego što je čovjek sposoban i promisliti o tome. Dakle, prema Gibsonu, slike su trenutačno u mozgu i tumače se kao stvarnost, dok po LeDouxu ljudi automatski reagiraju putem emocija. Ukratko, ljudski mozak je istreniran interpretirati slike kao stvarnost, a istovremeno reagirati putem emocija. Prema [16], činjenica da mozak obradom slike reagira emocijama podrazumijeva snažan utjecaj, koji javno dostupne slike imaju na društvo i pojedinca. Dr. Newton, sa Sveučilišta u Oregonu [16], navodi kako slike utječu na pamćene, ponašanje i vrijednosti. Slike utječu na sliku koju pojedinac ima o sebi, o drugima i o svijetu koji ga okružuje.

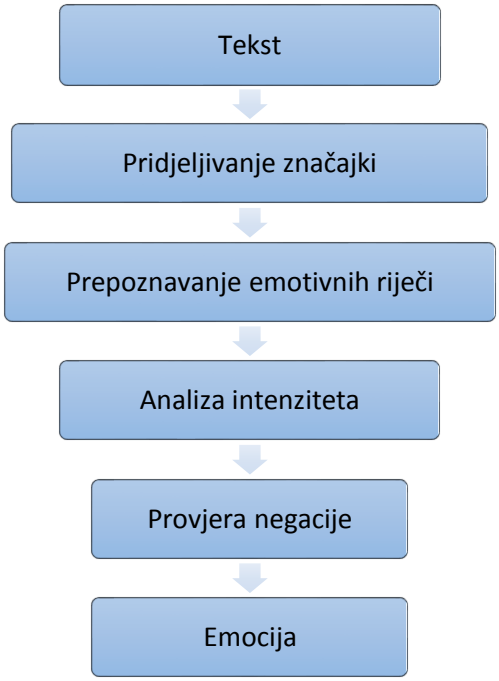
3.6. Utjecaj inspirativnih citata na pojedinca

Gotovo da i nema dostupne stručne literature koja se bavi utjecajem citata na emocije pojedinca. Međutim, mnoštvo je članaka dostupnih na Internetu, koji je ujedno i najveća baza citata, koji govore u njihovu korist. Tako prema [17], inspirativni citati jedan su od glavnih razloga uspjeha mnogo ljudi. Čitanje motivacijskih i nadahnjujućih citata, na dnevnoj bazi podiže motiviranost kod ljudi i pomaže im u poduzimanju onoga što žele. Smatra se kako čitanje inspirativnih citata, na dnevnoj bazi pomaže osobama da postanu uspješnije i sretnije. Postoje milijuni dostupnih citata iz raznih izvora, koji predstavljaju neiscrpan izvor motivacije. Kao osnovne prednosti čitanja citata, navodi se kako su oni brzi „podizači“ motivacije, pomažu u liječenje depresije, sprječavaju odugovlačenje kojem su pojedinci skloni te osim toga dostupni su u gotovo neograničenom broju. Ukratko, čitanje inspirativnih citata jedan je od najjednostavnijih, najbržih i najboljih načina za pozitivni utjecaj na emocije pojedinca.

4. METODE I PROBLEMI PREPOZNAVANJA EMOCIJA

4.1. Pretraga po ključnim riječima

Prema [18], ova metoda se u potpunosti oslanja na postojanost ključnih riječi u tekstu. Za nju je potrebno, prvenstveno, *parsiranje*, što zahtijeva puno resursa te je i vremenski zahtjevno. Zatim, potrebna je velika baza podataka ključnih riječi povezanih s emocijama koje one predstavljaju. Stvaranje takve baze zahtijeva puno vremena, a porastom količine podataka u bazi raste i pouzdanost ove metode. S druge strane, lako se implementira, intuitivna je te jednostavna za izvedbu i korištenje, jer se radi samo o traženju određenih ključnih riječi u tekstu. Metoda pretrage po ključnim riječima se sastoji od pet koraka, koje su prikazane na slici 4.1, gdje je ulaz neki tekst, a izlaz - klasa emocije.



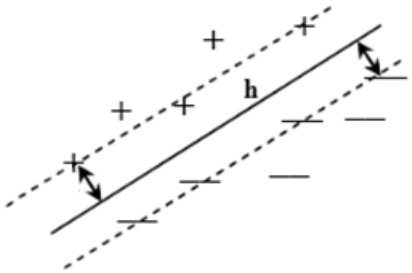
Slika 4.1. Dijagram toka metode prepoznavanja emocije po ključnim riječima

U prvom koraku se dijelovima teksta pridjeljuju značajke (engl. *tokenization*). U sljedećem koraku se pomoću tih značajki pronalaze i raspoznaju emocije. Nakon toga slijedi analiza intenziteta emocija u pronađenim ključnim riječima, koja obično varira između 0 i 1. Zatim slijedi provjera negacije u tekstu, kako bi se izbjeglo tumačenje negativnih emocija kao pozitivne, kao na primjer: „ne volim“,

„ne sviđa mi se“ i dr. U zadnjem koraku se računa ukupni intenzitet za pojedinu emociju te kao izlaz se šalje „najjača“ emocija.

4.2. Metoda na osnovi učenja

Prema [19], ova metoda koristi trenirani skup značajki, kako bi se ulazni tekst klasificirao na klase emocija koristeći ključne riječi kao značajke. Lakše i brže može se prilagoditi promjenama domene, budući da ona brzo može „naučiti“ nove značajke skupa treniranjem s novim, velikim skupom značajki za trening, koristeći algoritme za učenje te obnavljanjem klasifikacijskih modela. Iako, kreiranje velikog korpusa (skupa za treniranje) nije uvijek izvediv. Ipak, olakšava jednostavnu implementaciju klasifikatora i za početnike te mogu iskoristiti naučeni model i na drugim slučajevima, tekstovima. Glavni nedostatak ove metode je da dovodi do nejasnih granica između klasa emocija i nedostatka kontekstne analize. Kako bi klasifikacija emocija bila što uspješnija, koriste se razni algoritmi za učenje kao na primjer *SVM* (engl. *Support Vector Machine*) i *CRF* (engl. *Conditional Random Fields)*. Najbolje rezultate za ovu metodu ipak daje *SVM*, koja se bazira na statistikama. Postiže vrlo dobre rezultate performanse na raznim područjima (bio-informatika, prepoznavanje slike, teksta i dr.). *SVM* pretpostavlja simetričnu razdiobu između pozitivne i negativne granice (margine) s obzirom na središnje osi razdiobe, kako je prikazano na slici 4.2. Klasifikacija se temelji na udaljenosti margina od središnje razdiobe. Ova metoda se u radu koristi u procesu prepoznavanja emocija preko vanjskog poslužitelja (potpoglavlje 5.3.2).



Slika 4.2. Simetrična razdioba između pozitivne i negativne granice

4.3. Hibridne metode

S obzirom da metode prepoznavanja ključnih riječi s bazama podataka te metode na osnovi učenja ne daju zadovoljavajuće rezultate, često drugi sustavi za raspoznavanje emocija koriste kombinaciju tih dviju metoda, što prema [19] uvelike poboljšava rezultate i njihovu preciznost. Semantika i atributi se povezuju s emocijama, prema određenim pravilima povezivanja emocija. Kao rezultat tih pravila, zamjenjuju se izvorne emocije povezane s ključnim riječima te one služe kao novi skup za treniranje sustava za učenje. Ova metoda daje bolje rezultate u točnosti i zahtjeva manje resursa te vremenski se brže izvodi, ali kategorije emocija su još uvijek ograničene. S obzirom da je ova metoda najpreciznija upravo se ona koristi u ovome radu za prepoznavanje emocija iz napisanog teksta.

4.4. Problemi prepoznavanja emocija

Prema [19], postoji mnogo problema i ograničenja kod svih metoda za prepoznavanje emocija iz napisanog teksta što uvelike umanjuje i očekivanu točnost programskog rješenja ovoga rada. U sljedećim potpoglavljima navode se i opisuju tri glavna problema.

4.4.1. Višesmislenost definicija ključnih riječi

Korištenje ključnih riječi je jednostavan način prepoznavanja emocija, ali značenja ključnih riječi mogu biti različita i nejasna, budući da mnoge riječi mogu promijeniti svoja značenja ovisno o uporabi i kontekstu rečenice. Osim toga, čak i minimalan broj skupova ključnih riječi može označavati razne emocije u nekim ekstremnim slučajevima, kao na primjer kada se koriste ironične, sarkastične ili cinične rečenice.

4.4.2. Nemogućnost prepoznavanja emocija bez ključnih riječi

Metoda prepoznavanja emocija pomoću ključnih riječi u potpunosti se oslanja na postojanje ključnih riječi. Stoga bi se podrazumijevalo da su rečenice bez ijedne ključne riječi emotivno neutralne, odnosno, da ne postoji niti jedna emocija u njima, što ne mora biti slučaj. Ručno određivanje vrijednosti za svaku pojedinu riječ iz rječnika bi zahtijevalo puno ljudskih resursa te s time veće troškove, ali bi i sustav bio puno sporiji pronalazeći svaku pojedinu riječ iz rečenice u velikoj listi te računajući vrijednosti za pojedinu emociju.

4.4.3. Problem nedostatka jezične informacije

Struktura sintakse i semantike, također, ima utjecaja na izraženu emociju. Na primjer, rečenice "Smijao sam mu se" i "Smijao mi se" sadrže iste ključne riječi, ali emocije koje predstavljaju su potpuno različite. Kao rezultat ignoriranja jezične informacije, metode bazirane na ključnim riječima bi rečenicama mogle pridijeliti iste emocije, iako, zapravo mogu biti i potpuno suprotne.

5. TEHNOLOGIJE KORIŠTENE PRI IZRADI APLIKACIJE EMODAPTER

5.1. Operacijski sustav Android

Prema [20], operacijski sustav (OS) Android koristi stotine milijuna mobilnih uređaja u više od 190 država svijeta. Temeljen je na tzv. Linux platformi otvorenog koda, koji već dugi niz godina konstantno proširuje svoj krug programera, koji redovito pridonose ubrzanom razvoju sustava. Kako je Android naslijedio svoj osnovni kod iz Linuxa, također, je naslijedio i njegove programere te danas već i preko 300 velikih partnera proizvođača sklopovlja, računalnih programa te ulagače. Tako je Android postao najbrže rastući mobilni operacijski sustav.

Sve dosadašnje inačice Androida imaju naziv po nekom slatkišu te svaki počinje sa sljedećim slovom u engleskoj abecedi. Trenutna inačica je „Marshmallow“, ali najveći broj korisnika još uvijek koristi inačice „KitKat“ i „Lolipop“.

Android pruža sve što programeru zatreba za brzu i kvalitetnu izradu aplikacija, prilagođenu za bilo koji mobilni uređaj pogonjen OS-om Android. Također, pruža alate za izradu grafičkog sučelja te pristup svim ugrađenim ili povezanim sklopovljima, što omogućuje dobar izgled aplikacije, uz potpunu kontrolu sučelja, a uz to i automatsko prilagođavanje na pojedinom uređaju. Kako bi se izrada aplikacije još više ubrzala i pojednostavila izdan je i službeni alat za izradu, popravak i izdavanje Android aplikacije u Java programskom jeziku – Android Studio, koji će biti opisan u slijedećem poglavlju.

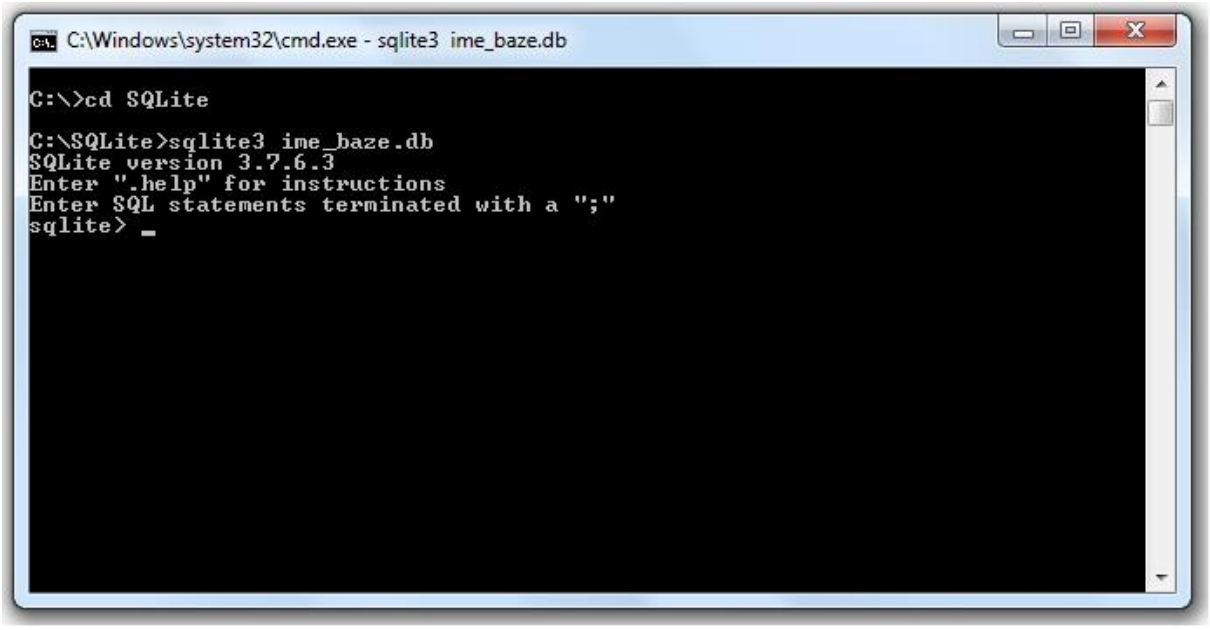
5.1.1. Android Studio

Prema [21], Android Studio je službeno integrirano razvojno okruženje (engl. *Integrated Development Environment - IDE*) za razvoj aplikacija, čija je podloga proizvod „IntelliJ IDEA“ tvrtke „JetBrains“. Uz snažan uređivač koda i razvojne alate, Android Studio pruža i mnoge druge mogućnosti koje pridonose produktivnosti pri izradi Android aplikacija, kao na primjer: podrška za C++ i NDK (Native Development Kit), Lint alate za testiranje performansi, kompatibilnosti i ostalih problema, alate za testiranje itd.

U radu se koristi za izradu virtualne tipkovnice za OS Android, praćenje unošenog teksta korisnika te promjene na grafičkom sučelju.

5.2. SQLite

Prema [22], SQLite je knjižnica koja sadržava SQL procedure za baze podataka. Za razliku od većine SQL baza podataka SQLite nema odvojenih poslužiteljskih procesa, već sve tablice, podatke i ostale elemente izravno čita i piše u običan dokument na tvrdom disku. Taj dokument je potpuno prenosiv neovisno o operacijskom sustavu, 32- ili 64- bitnom sustavu te različitim arhitekturama računala. Također, vrlo je kompaktan te veličina baze podataka sa svim uključenim mogućnostima može biti i manja od 500kB. Zbog toga je i vrlo brza te je brža kada ima više radne memorije na raspolaganju.
SQLite je vrlo pažljivo testiran na milijunima test slučajeva sa stotinama milijuna zasebnih *SQL* naredbi te je postigao rezultat od 100% pokrivenosti testnih slučajeva, tako da s velikom sigurnošću možemo reći da je također i vrlo pouzdan. Štoviše, otporan je i na „pad“ sustava ili nestanak struje pri radu s bazom podataka. Naravno ni jedan sustav ne može biti savršen pa tako postoji lista pogrešaka, tzv. „buba“ (engl. *bug*), ali kao i Android – SQLite je otvorenog koda te veliki krug programera rade na unapređivanju sustava i uklanjanju grešaka.
Uz to sve nije joj potrebno ni početno postavljanje, već je potrebno samo preuzeti izvršnu datoteku te jednostavno u sučelju za upravljanje komandnim linijama pozvati datoteku i kao argument dodati ime baze, odnosno datoteke, kao što je vidljivo na slici 5.1.



Slika 5.1. Primjer korištenja SQLite u komandnom uređivaču

Iz ovih i još mnogo drugih razloga potpuno je jasno zašto Android ima ugrađene klase i metode za rad sa SQLite bazom podataka.
U radu se SQLite koristi za izradu baze podataka za pohranu inspirativnih citata te testnih podataka.

5.3. API

Aplikacijsko programsko sučelje (engl. *Application Programming Interface*, API) je skup naredbi i alata za izradu aplikacija. Omogućuje komunikaciju između dvije aplikacije ili korištenje dijela vanjske aplikacije slanjem i primanjem podataka u određenom obliku, a najčešće u JSON obliku.

5.3.1. API Yandex Translate

Yandex je jedna od najvećih Internet tvrtki u Europi te vodi najpopularniju tražilicu u Rusiji. Slično kao i Google, imaju svoju inačicu tražilice, lokacijske karte, analize prometa i podataka, prepoznavanje glasovnih naredbi i još mnoge druge API-ove među kojima je i Translate API za prevođenje teksta.
Prema [23], Translate API pruža pristup mrežnoj usluzi za prevođenje te podržava više od 70 jezika na kojima može prevesti zasebne riječi ili cijele rečenice. Za rad s ovim API-em potreban je besplatan ključ za koji se treba registrirati na njihovoj stranici. Uz API, postoji i detaljna dokumentacija, koja uvelike olakšava implementaciju u vlastitu aplikaciju. Tekst koji se želi prevesti može se poslati kao POST ili GET zahtjev HTTPS protokola, što pruža sigurnu komunikaciju između aplikacije i poslužitelja. Prevedeni tekst i drugi parametri s poslužitelja se mogu zatražiti u tri oblika: XML (*Extensible Markup Language*), JSON (*JavaScript Object Notation*) ili JSONP.
XML je jezik za obilježavanje definiran od strane W3C, a njegov format se može otvoriti bilo kojim tekstualnim uređivačem te je potpuno čitljiv i razumljiv ljudima. Za označavanje se koriste oznake (engl. *tag*) kao i u HTML-u. JSON je sintaksa za spremanje i prenošenje podataka u obliku objekata i nizova. Iako, koristi JavaScript sintaksu, također, je čitljiv i razumljiv ljudima, ali je neovisan o programskom jeziku te je kompaktniji od XML-a, što smanjuje promet pri prenošenju iste količine podataka. Primjer povratne informacije iz Translate API-a u JSON obliku je vidljiv na slici 5.2.

```
{
    "code":200,
    "lang":"hr-en",
    "text":[
        "An example of the translated text"
    ]
}
```


Slika 5.2. Primjer povratne informacije u *JSON* obliku

JSONP je mehanizam koji izravno zahtjeva podatke s poslužitelja putem HTML <*script*> oznake u JSON obliku dodajući parametar *callback* i ime funkcije koja vraća JSON. Omogućuje smanjenje podataka za slanje te time i promet i vrijeme izvršavanja aplikacije. Postoje novija i efektivnija rješenja, ali se JSONP još uvijek koristi zbog podržavanja starijih inačica mrežnih preglednika. Popis izlaznih vrijednosti i opisa za Translate API se prema [23] nalazi su tablici 5.1.

Tablica 5.1. Popis izlaznih vrijednosti i opisa

Vrijednost	Opis
200	Operacija uspješno izvršena
401	Pogrešan API ključ
402	Blokirani API ključ
404	Dostignut dnevni limit količine prevedenog teksta
413	Prijeđena maksimalna duljina teksta
422	Tekst se ne može prevesti
501	Zadani smjer prijevoda nije moguć

API Yandex se u radu koristi za prevođenje teksta sa hrvatskog na engleski prije prepoznavanja emocija s API Tone Analyzer, koji će biti opisan u potpoglavlju 5.3.2. Tablica 5.1. je bitna za uspješan rad aplikacije. Ako je vraćena vrijednost bilo koja osim „200“ to znači da se negdje dogodila greška i da aplikacija može prestati radit ili krivo prepoznati emocije. U tom slučaju se brže pronalazi i ukljanja greška poznavajući opis svakog pojedinog izlaznog koda.

5.3.2. API Tone Analyzer

Prema [24], Tone Analyzer je usluga, koji jezičnom analizom napisanog teksta omogućuje prepoznavanje emocija, društvenih sklonosti i stila pisanja te podržan je od strane IBM-a. Bazirana je na teoriji psiholingvistike, odnosno polja koja istražuje poveznice između jezičnih ponašanja i psiholoških teorija. Točnije, istražuju jesu li riječi koje koristimo u svakodnevnom životu zrcalo onoga tko smo mi, kako se osjećamo te kako razmišljamo, odnosno, naše osobnosti. Nakon nekoliko desetljeća istraživanja psiholingvistika je priznata u psihologiji, marketingu i ostalim područjima, gdje riječi znače više od onoga što želimo reći. Učestalost s kojom koristimo određene tipove riječi može dati trag osobnosti, stila razmišljanja, društvenih veza i emocionalnog stanja. Upravo na tim načelima je Tone Analyzer utemeljen.

Usluga računa jezične tonove pomoću jezične analize, koja promatra veze između različitih tonova i jezičnih značajki u napisanom tekstu. API kao rezultat vraća tri glavne skupine jezičnih tonova: emocionalni, društveni i jezični. Za dobivanje emocionalnih tonova koristi se hibridna metoda prepoznavanja emocija, opisana u poglavlju 4.3. Emocije su kategorizirane prema Ekmanovoj raspodjeli, s izuzetkom emocije iznenađenja, te za svaku od njih API vraća vrijednost između 0 i 1, koja predstavlja koeficijent postojanja emocije u tekstu. Društveni tonovi su: otvorenost, emocionalnost, prihvaćenost, ekstraverzija i osviještenost. Jezični tonovi su: analitički, samopouzdani i privremeni. U radu se koriste samo emocionalni tonovi.

Točnost prepoznavanja emocija se kreće između 41% i 68%. Unatoč ovim malim brojevima te činjenici da se ovo područje istražuje već desetljećima, Tone Analyzer je jedan od najtočnijih sustava za prepoznavanje emocija. Iz tih razloga nije besplatan, ali dozvoljava probno korištenje do 30 dana od dana registracije.

U radu se koristi isključivo za prepoznavanje emocija, nakon što se pomoću API Translate (potpoglavlje 5.3.1) tekst prevede sa hrvatskog na engleski.

5.4. PHP

Prema [25], PHP (engl. *Hypertext Preprocessor*) je skriptni jezik, koji se ugrađuje u *HTML* kod. Sintaksa je većinom naslijeđena iz C, Java i Perl programskih jezika, ali ima i puno jedinstvenih značajki, koji su specifični samo za PHP. Pri kreiranju ovog jezika glavni cilj je bio omogućiti programerima brzo razvijanje dinamičkih mrežnih stranica. PHP skripte se izvršavaju na

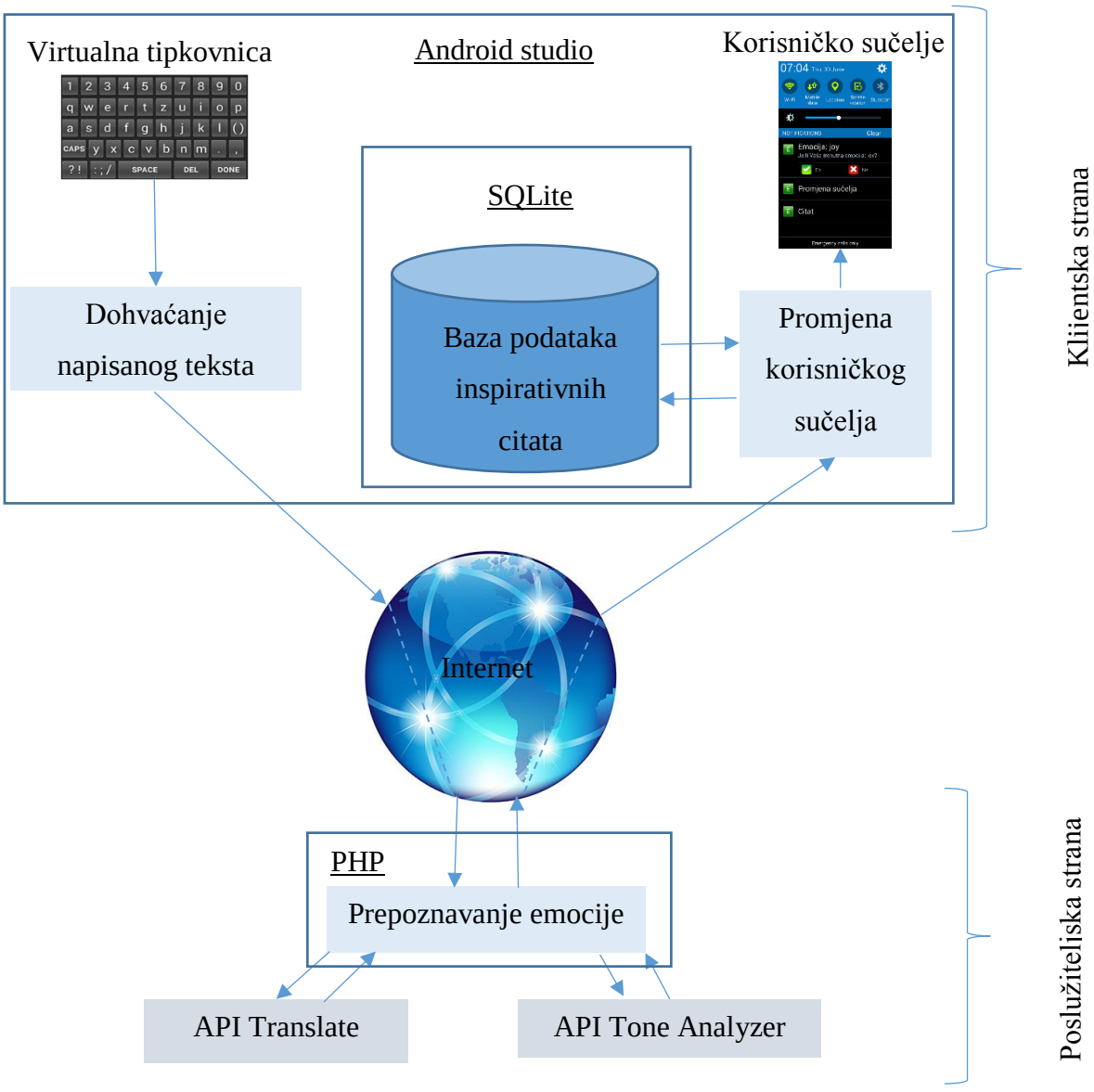
poslužiteljima, gdje su one spremljene te krajnji rezultat se šalje korisniku putem mrežnog preglednika. Tako korisnici nemaju uvid u procese koje se događaju pri otvaranju mrežne stranice. PHP skripte su lako prepoznatljive po njihovoj ekstenziji: „.php“. Mogu sadržavati čisti HTML kod ili dinamički ispisivati HTML, ili oboje. Za označavanje početka PHP skripte se koristi izraz „<?php“ (ili u novijim inačicama samo „<?“), a za kraj „?>“. Velike prednosti PHP skriptnog jezika su jednostavnost i mnoštvo ugrađenih i testiranih funkcija, koja početnicima omogućuje brzo učenje, a profesionalcima brzo razvijanje mrežnog sučelja uz malo koda i lake čitljivosti. Isto tako, još jedna velika prednost je i brzina, jer osnova PHP-a je C programski jezik, koji je jedan od „nižih“ programskih jezika, odnosno bliže je strojnom kodu od ostalih. Također, PHP podržava i kreiranje zasebnih aplikacija bez ili s grafičkim sučeljem, bez potrebe za poslužiteljem. PHP skripte bez grafičkog sučelja se najčešće koriste za radnje koje se trebaju ponavljati u određenom vremenskom periodu, kao na primjer obrada teksta ili obrada unosa u bazama podataka te se one pozivaju preko komandne linije, odnosno postavljanjem poziva koristeći *cron* na Linux ili *Task Scheduler* na Windows operacijskom sustavu. PHP nije najbolji jezik za razvoj aplikacija s grafičkim sučeljem, ali zbog broja ugrađenih funkcija, lakom pristupu komponentama sustava te neovisnosti o sustavu, ipak, nekad može biti i najbolje rješenje. PHP, također, podržava proceduralno i objektno orijentirano programiranje te njihovu kombinaciju.

PHP kao poslužiteljska aplikacija nije ograničena samo na HTML, već ima mnoštvo funkcija za rad sa slikama, raznim tekstualnim formatima, PDF datotekama pa čak i Flash videima te za njihov ispis u klijentovom mrežnom pregledniku. Ipak, jedna od najznačajnijih funkcija PHP je podrška za širok krug baza podataka te jednostavnost rada s njima, koristeći neki od raznih klasa i slojeva, kao na primjer: mysqli, PDO, ODBC itd.

PHP skripte se u programskom rješenju koriste za prikupljanje inspirativnih citata i testnih podataka te komunikaciju između Android aplikacije i API-a.

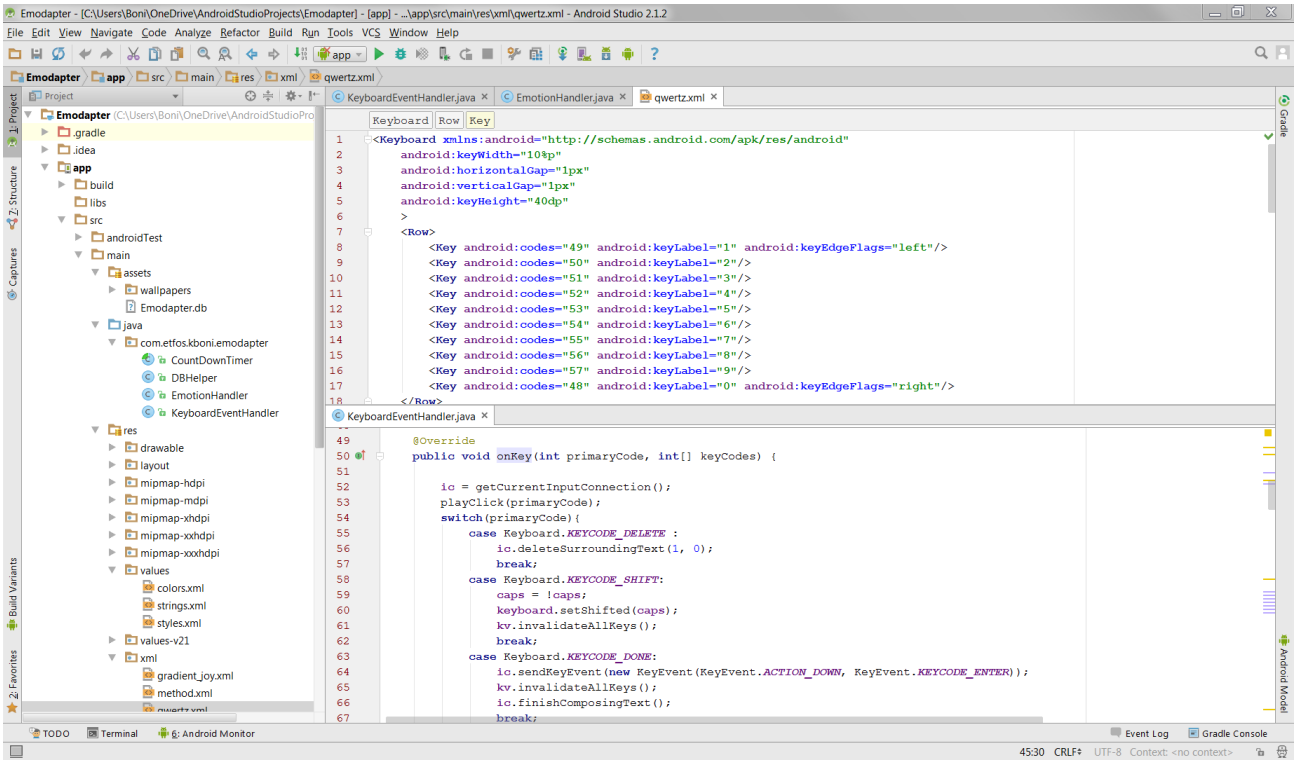
6. PROGRAMSKO RJEŠENJE APLIKACIJE EMODAPTER

Programsko rješenje, odnosno aplikacija nazvana „Emodapter“, najvećim dijelom je razvijeno u okruženju Android Studio, opisano u poglavlju 5.1.1, u programskom jeziku Java i opisnom jeziku XML. Programski jezik Java je korišten za izradu za tzv. engl. *backend*-a, to jest dio aplikacije koji korisniku nije vidljiv, a opisni jezik XML korišten je za opis izgleda pojedinih dijelova sučelja. Na slici 6.1 prikazana je struktura cijelog programskog rješenja.



Slika 6.1. Programska struktura programskog rješenja

Na slici 6.2, prikazano je razvojno okruženje Android Studio te unutar toga, s lijeve strane - stablo datoteka projekta, a s desne strane, gore, isječak XML koda, a dolje isječak Java koda.



Slika 6.2. Razvojno okruženje Adnroid Studio s primjerom Java i XML koda

6.1. Virtualna tipkovnica

Osnovni dio aplikacije je virtualna tipkovnica, tzv. engl. *soft keyboard*, s tipkama posloženim prema standardnoj hrvatskoj tipkovnici, odnosno modelu QWERTZ. Za njenu izradu koristi se klasa *Java* proširena klasom *InputMethodService*, koja implementira sučelje *OnKeyboardActionListener* iz klase *KeyboardView*. Klasa *InputMethodService* sadrži rukovatelje za osnovne elemente za unos podataka i metode za upravljanje njima. U ovom slučaju ona je poveznica između virtualne tipkovnice i bilo kojeg elementa za unos teksta, a isto tako služi i za čitanje trenutnog sadržaja u bilo kojem aktivnom elementu za unos teksta. Sučelje *OnKeyboardActionListener* služi za praćenje svih događaja vezanih za virtualnu tipkovnicu, kao na primjer pritisak na tipku, otpuštanje tipke, povlačenje prstom po tipkovnici u različitim smjerovima itd. Kako je prikazano u programskom kodu 6.1, ukupno je osam

apstraktnih metoda koje se moraju implementirati kako kodni prevoditelj (engl. *compiler*) ne bi javio greške: *onKey*, *onPress*, *onRelease*, *onText*, *swipeDown*, *swipeLeft*, *swipeRight* i *swipeUp*.

```
public class Example extends InputMethodService
        implements KeyboardView.OnKeyboardActionListener
{
    @Override
    public void onPress(int primaryCode){
    }
    @Override
    public void onRelease(int primaryCode){
    }

    @Override
    public void onKey(int primaryCode, int[] keyCodes){
    }

    @Override
    public void onText(CharSequence text){
    }

    @Override
    public void swipeLeft(){
    }

    @Override
    public void swipeRight(){
    }

    @Override
    public void swipeDown(){
    }

    @Override
    public void swipeUp(){
    }
}
```

Programski kod 6.1. Primjer klase za rad s virtualnom tipkovnicom

Nadalje, koriste se tri osnovne XML datoteke. Prva od njih (programski kod 6.2) definira podatke koje se prikazuju korisniku pri odabiru virtualne tipkovnice koju će koristiti u svim aplikacijama.

```
<resources>
    <string name="app_name">Emodapter</string>
    <string name="simple_ime">Emodapter</string>
    <string name="subtype_hr_HR">Croatian (HR)</string>
</resources>
```

Programski kod 6.2. XML kod za definiranje osnovnih podataka aplikacije

Druga datoteka (programski kod 6.3) definira osnovne značajke izgleda tipkovnice.

```
<?xml version="1.0" encoding="UTF-8"?>
<android.inputmethodservice.KeyboardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/keyboard"
    android:layout_width="match_parent"
```

```
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    />
```

Programski kod 6.3. XML kod za definiranje izgleda tipkovnice

Treća XML datoteka definira položaj i izgled tipki te dekadsku vrijednost i svojstva svake pojedine tipke, a kod ove datoteke nalazi se u P.6.1. Navedene datoteke su samo osnovne datoteke za početak izrade aplikacije, što znači da se u projektu nalazi još mnoštvo datoteka koje su potrebne za cjelovitu aplikaciju.

6.2. Dohvaćanje napisanog teksta

Sljedeći važan dio aplikacije je dohvaćanje napisanog teksta korisnika. Većina aplikacija ne podržava mogućnost slanja napisane poruke pritiskom na tipku na virtualnoj tipkovnici, već za tu funkciju koriste svoje ugrađene tipke unutar vlastite aplikacije. Poveznica između virtualne tipkovnice i druge aplikacije u koju se unosi tekst je tekstni okvir aplikacije. U programskom kodu ova poveznica je, već spomenuta, Java klasa *InputMethodService* pomoću čijih metoda se prate događaji unutar tekstnog okvira te reagiraju na njih. Njena najvažnija metoda je *onKey*, koja prati svaki pritisak na tipku te definira događaj za svaku pojedinačnu tipku, kao na primjer brisanje teksta, upis teksta, pisanje velikim slovima itd. Kao parametre ima dekadski broj pojedine tipke te listu dekadskih brojeva svih tipki, koje su definirane u već spomenutoj XML datoteci iz P.6.1.

Pritisak na tipku za slanje poruke unutar aplikacije poziva metodu *onStartInputView*, što upućuje na to da slanjem poruke aplikacija ponovno poziva virtualnu tipkovnicu. Premošćenjem (engl. *override*) ove funkcije dohvaća se „znak“ da je poruka poslana, međutim pomoću nje se ne može dohvatiti napisani tekst jer u trenutku kada „znak“ stigne, poruka je poslana, odnosno, tekstualni okvir je prazan. Ovaj problem je riješen dohvaćanjem upisanog teksta svakim pritiskom na tipku na virtualnoj tipkovnici u metodi *onKey*. Završeni kod ovoga dijela nalazi se u slijedećem programskom kodu:

```
@Override
public void onKey(int primaryCode, int[] keyCodes) {

    ic = getCurrentInputConnection();
    playClick(primaryCode);
    switch(primaryCode){
        case Keyboard.KEYCODE_DELETE :
            ic.deleteSurroundingText(1, 0);
            break;
        case Keyboard.KEYCODE_SHIFT:
            caps = !caps;
            keyboard.setShifted(caps);
```

```
            kv.invalidateAllKeys();
            break;
        case Keyboard.KEYCODE_DONE:
            ic.sendKeyEvent(new KeyEvent(KeyEvent.ACTION_DOWN, KeyEvent.KEYCODE_ENTER));
            kv.invalidateAllKeys();
            ic.finishComposingText();
            break;
        default:
            char code = (char)primaryCode;
            if(Character.isLetter(code) && caps)
            {
                code = Character.toUpperCase(code);
            }
            ic.commitText(String.valueOf(code),1);
            try
            {
                this.text = ic.getExtractedText(new ExtractedTextRequest(), 0).text;
            }
            catch(Exception e)
            {
                Log.d("Text extraction", e.getMessage());
            }
            break;
    }

}
```

Programski kod 6.4. Java kod metode *onKey* unutar klase *KeyboardEventHandler*

Kako bi se smanjio mrežni podatkovni promet i opterećenje poslužitelja, ne šalje se svaka napisana rečenica zasebno, već se u metodi *onStartInputView* svaki poslani tekst sprema u listu nizova znakova, te ako ukupni broj znakova prijeđe 500, onda se poziva metoda za prepoznavanje emocija, kako je prikazano u sljedećem kodu:

```
@Override
public void onStartInputView(EditorInfo attribute, boolean restarting) {
    super.onStartInputView(attribute, restarting);

    if(this.text.length() > 0)
    {
        textLength += this.text.length();
        this.textList.add(this.text.toString());
    }

    if(this.textLength > 500)
    {
        new getPostEmotion().execute(TextUtils.join(". ", this.textList));
        this.text = "";
        this.textList.clear();
    }
}
```

Programski kod 6.5. Java kod metode *onStartInputView* unutar klase *KeyboardEventHandler*

Kada korisnik završi s pisanjem teksta te „zatvori“, odnosno sakrije virtualnu tipkovnicu, poziva se metoda *onFinishInputView* u kojoj se vrše provjere postojanja teksta u listi, spajanje tekstova u jedan tekst kao zasebne rečenice te pozivanje metode za prepoznavanje emocije (programski kod 6.6).

```
@Override
public void onFinishInputView(boolean finishingInput) {
    super.onFinishInputView(finishingInput);

    int listSize = this.textList.size();
    if(this.text.length() > 0)
    {
        String t = this.text.toString();
        String lastText = listSize > 0 ? this.textList.get(listSize - 1) : "";
        if(lastText != t)
        {
            this.textList.add(t);
        }
    }

    if(listSize > 0)
    {
        new getPostEmotion().execute(TextUtils.join(". ", this.textList));
        this.text = "";
        this.textList.clear();
    }
}
```

Programski kod 6.6. Java kod metode *onFinishInputView* unutar klase *KeyboardEventHandler*

6.3. Prepoznavanje emocije

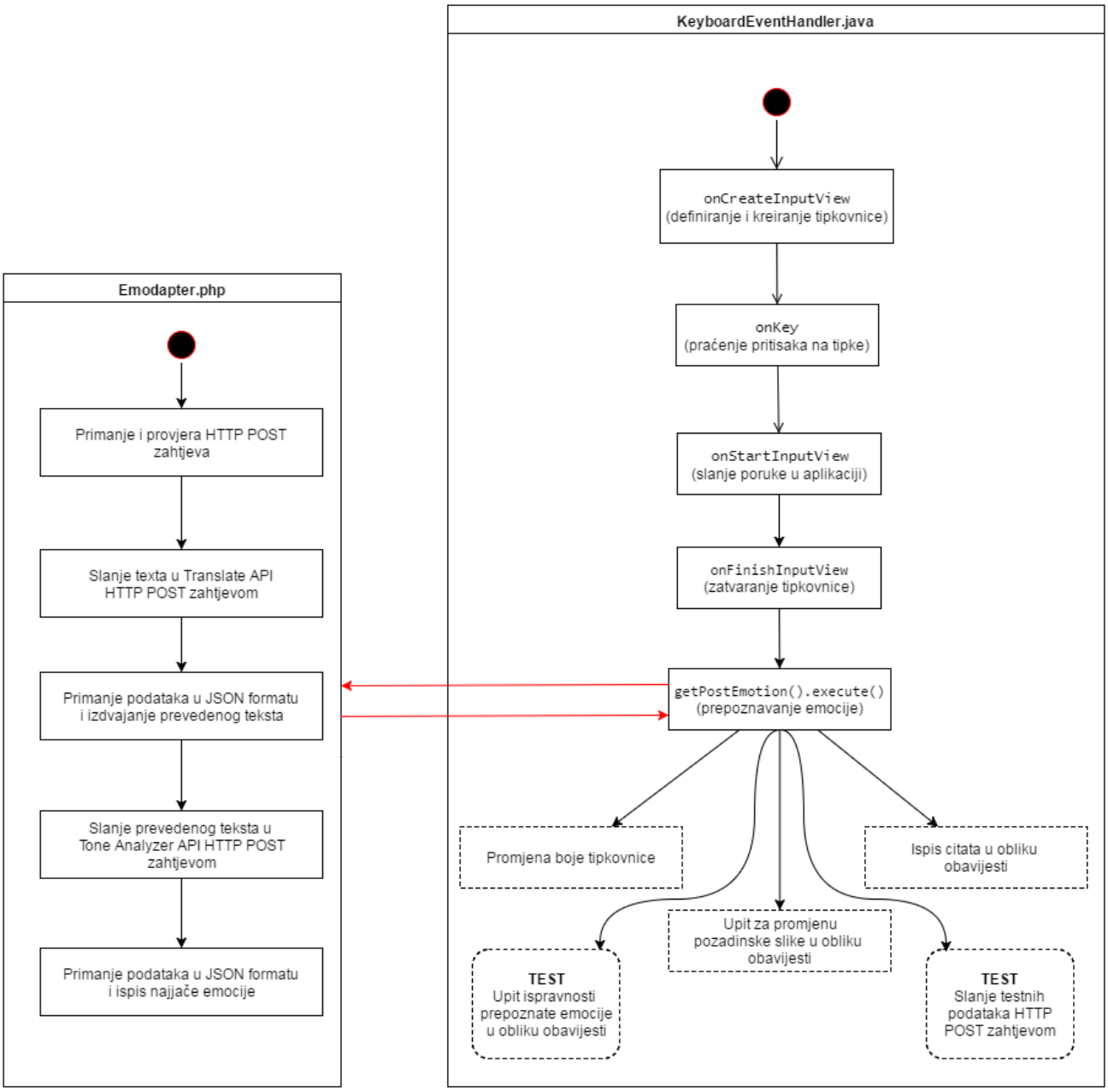
Klasa za prepoznavanje emocije (programski kod 6.7), *getPostEmotion*, izvršava se asinkrono, odnosno „u pozadini“ s klasom *AsyncTask*, kako aplikacija ne bi uzrokovala zastoj pri korištenju tipkovnice. Ovaj dio se sastoji od dvije metode: *doInBackground* i *onPostExecute*. U *doInBackground* šalje se tekst prema PHP skripti kao HTTP POST zahtjev, a u *onPostExecute* čeka se „odgovor“ koji sadrži prepoznatu emociju te na osnovi prepoznate emocije pozivaju se metode o kojima će kasnije biti riječi.

```
public class getPostEmotion extends AsyncTask<String, String, String>
{
    @Override
    protected String doInBackground(String... args) {

        if(args[0].isEmpty())
        {
            return "";
        }
        String emotion = "";
        String text;
        try
        {
            text = args[0];
```

```
            URL url = new URL("http://kboni.site40.net/diplomski/emodapter.php");
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();

            conn.setReadTimeout(10000);
            conn.setConnectTimeout(15000);
            conn.setRequestMethod("POST");
            conn.setDoInput(true);
            conn.setDoOutput(true);

            Uri.Builder builder = new Uri.Builder()
                    .appendQueryParameter("text", text);
            String query = builder.build().getEncodedQuery();

            OutputStream os = conn.getOutputStream();
            BufferedWriter writer = new BufferedWriter(
                    new OutputStreamWriter(os, "UTF-8"));
            writer.write(query);

            writer.flush();
            writer.close();
            os.close();
            int responseCode=conn.getResponseCode();
            if (responseCode == HttpURLConnection.HTTP_OK) {
                String line;
                BufferedReader br=new BufferedReader(new InputStreamReader(conn.getInputStream()));

                if((line=br.readLine()) != null)
                {
                    emotion=line;
                }
            }
        }catch( Exception e)
        {
            Log.v("Async Error", e.getMessage());
        }
        return emotion.trim();
    }
@Override
    protected void onPostExecute(String emotion) {
        for (String e:emotions)
        {
            if(emotion.equalsIgnoreCase(e))
            {
                EmotionHandler.setEMOTION(emotion);
                emotionHandler.quoteNotification(getApplicationContext(), emotion);
                setInputView(initializeSoftKeyboard(emotion));
                questionNotification();
                break;
            }
        }
    }
}
```

Programski kod 6.7. Java kod asinkrone klase *GetPostEmotion* unutar klase *KeyboardEventHandler*

Dijagram toka glavnog dijela aplikacije nalazi se na slici 6.2.



Slika 6.2. Dijagram toka glavnog dijela aplikacije Emodapter

PHP datoteka sadrži dvije metode: *translate* i *getEmotion*, a nalazi se na besplatnom poslužitelju. Pri primanju zahtjeva s tekstom prvo se provjerava je li tip HTTP zahtjeva POST te postoji li tekst. Nakon toga, poziva se metoda *translate* koja tekst, istim zahtjevom, šalje prema Yandex Translate API-ju, opisan u poglavlju 5.3.1. Odgovor stiže u JSON obliku te sadrži različita polja od kojih je u ovom slučaju potreban samo tzv. izlazni kod (engl. *exit code*), koji govori je li prevođenje bilo uspješno, te

prevedeni tekst. Ako je vraćeni kod 200, tj. tekst je uspješno preveden, onda se poziva metoda *getEmotion* koja prevedeni tekst šalje na Tone Analyzer API, također HTTP POST zahtjevom. Nakon analize teksta API vraća informacije navedene u poglavlju 5.3.2. Poslije toga se prolazi kroz sve vrijednosti za pojedinu emociju te ispisuje emocija s najvećom vrijednosti. Ta emocija se tada dohvaća u Android aplikaciji. Kod PHP skripte nalazi se u P.6.2.

6.4. Promjena korisničkog sučelja

Nakon dohvaćanja emocije, u već spomenutoj klasi *getPostEmotion,* u metodi *onPostExecute* pozivaju se metode *initializeSoftKeyboard*, *setInputView* i *questionNotification* te iz klase *EmotionHandler* metoda *quoteNotification*. U metodi *initializeSoftKeyboard* se učitavaju postavke za izgled tipkovnice za prepoznatu emociju iz pojedine XML datoteke, kako je prikazano u kodu:

```
private View initializeSoftKeyboard(String emotion)
{
    int layout;
    switch(emotion)
    {
        case "anger":
            layout = R.layout.keyboard_anger;
            break;
        case "disgust":
            layout = R.layout.keyboard_disgust;
            break;
        case "fear":
            layout = R.layout.keyboard_fear;
            break;
        case "joy":
            layout = R.layout.keyboard_joy;
            break;
        case "sadness":
            layout = R.layout.keyboard_sadness;
            break;
        default:
            layout = R.layout.keyboard;
            break;
    }
    kv = (KeyboardView)getLayoutInflater().inflate(layout, null);
    keyboard = new Keyboard(this, R.xml.qwertz);
    kv.setKeyboard(keyboard);
    kv.setOnKeyboardActionListener(this);

    return kv;
}
```

Programski kod 6.8. Java kod metode *initializeSoftKeyboard* unutar klase *KeyboardEventHandler*

Virtualna tipkovnica se ponovo poziva pomoću metode *setInputView:*

```
setInputView(initializeSoftKeyboard(emotion));
```

Programski kod 6.9. Java kod poziva metoda za ponovno postavljanje virtualne tipkovnice

Primjer XML koda postavki za emociju „radost“:

```
<?xml version="1.0" encoding="UTF-8"?>
<android.inputmethodservice.KeyboardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/keyboard"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:keyTextColor="@color/joy_yellow_1"
    android:keyBackground="@color/joy_blue_3"
    android:background="@color/white"
    android:shadowRadius="0.0"
    />
```

Programski kod 6.10. XML kod za postavke izgleda tipkovnice za emociju „radost“

Metoda *questionNotification* (programski kod 6.11) postavlja obavijest (engl. *notification*) kako bi korisnik mogao odlučiti želi li promjenu pozadinske slike. Vrijednosti tipaka za odabir su unaprijed definirani u *AndroidManifest.xml* datoteci koja sadrži najznačajnije postavke aplikacije, kao npr. inačica Androida za koju se aplikacija radi, dozvole korištenja raznih dijelova uređaja itd. Pritiskom na tipku „Da“ ili „Ne“ šalje se asinkroni zahtjev prema klasi *EmotionHandler* koji signal prima pomoću proširenja s klasom *BroadcastReceiver* te premošćenjem metode *onReceive*. Kod metode *questionNotification*:

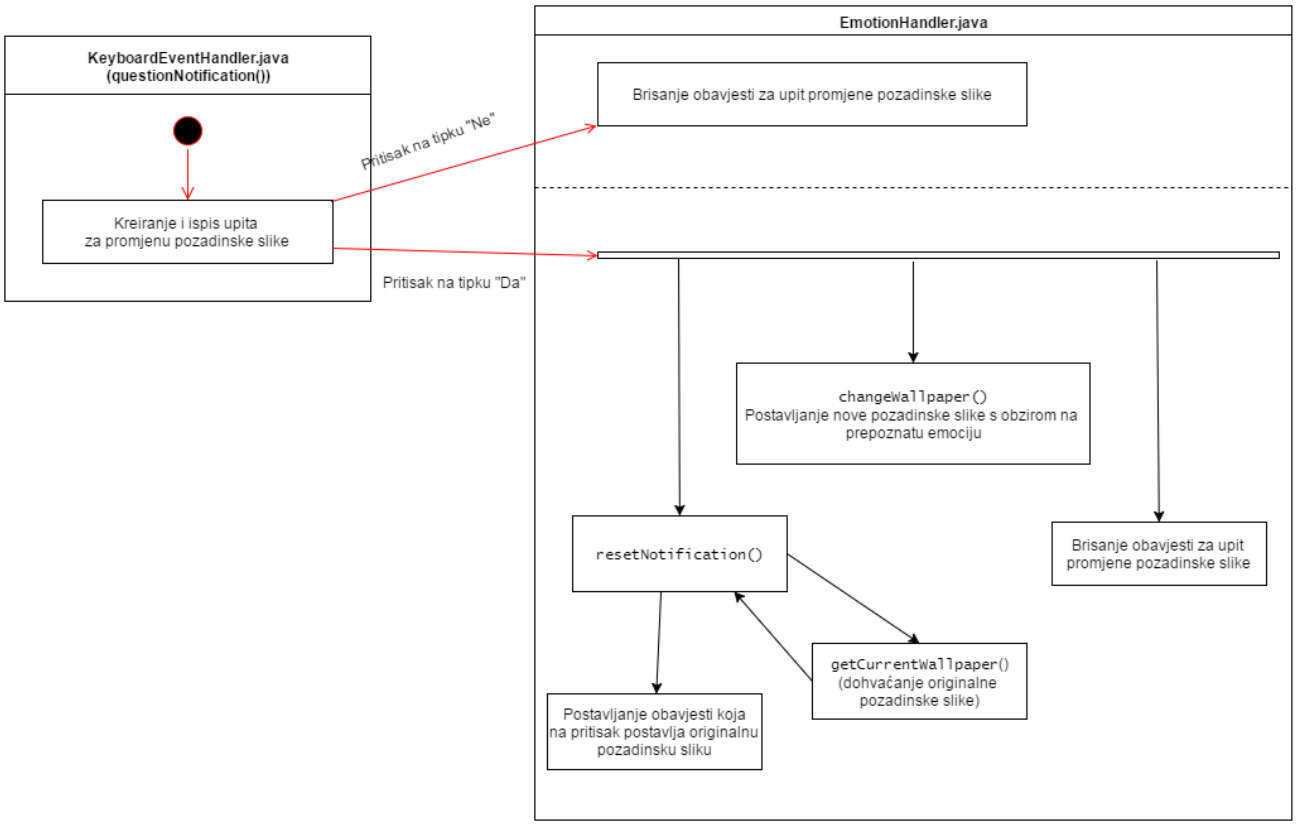
```
public void questionNotification()
{
    String question = "Dozvoljavate li promjenu pozadine na osnovi vašeg emotivnog stanja?";
    Intent yes_intent = new Intent(getApplicationContext(), EmotionHandler.class);
    yes_intent.setAction(EmotionHandler.YES_ACTION);
    Intent no_intent = new Intent(getApplicationContext(), EmotionHandler.class);
    no_intent.setAction(EmotionHandler.NO_ACTION);

    PendingIntent yes_pIntent = PendingIntent.getBroadcast(getApplicationContext(),
          (int) System.currentTimeMillis(), yes_intent, PendingIntent.FLAG_UPDATE_CURRENT);
    PendingIntent no_pIntent = PendingIntent.getBroadcast(getApplicationContext(),
          (int) System.currentTimeMillis(), no_intent, PendingIntent.FLAG_UPDATE_CURRENT);

    NotificationCompat.Builder mBuilder =
            new NotificationCompat.Builder(getApplicationContext())
                    .setSmallIcon(EmotionHandler.notificationIcon)
                    .setContentTitle("Promjena sučelja")
                    .setStyle(new NotificationCompat.BigTextStyle().bigText(question))
                    .addAction(R.drawable.ok, "Da", yes_pIntent)
                    .addAction(R.drawable.cancel, "Ne", no_pIntent)
                    .setContentIntent(PendingIntent.getActivity(getApplicationContext(), 0,
                                      new Intent(), 0))
                    .setAutoCancel(true)
                    .setWhen(0).setPriority(Notification.PRIORITY_MAX);
    NotificationManager mNotificationManager =
            (NotificationManager)
getApplicationContext().getSystemService(Context.NOTIFICATION_SERVICE);
    mNotificationManager.notify(EmotionHandler.questionNotificationId , mBuilder.build());
}
```

Programski kod 6.11. Java kod metode *questionNotification* unutar klase *KeyboardEventHandler*

Metoda *onReceive* (programski kod 6.14) se sastoji od uvjeta *switch-case* koji ima tri slučaja: dva navedena, *YES_ACTION* i *NO_ACTION*, te *RESET_ACTION* o kojem će biti riječi kasnije. Pritiskom na tipku „Ne“ u obavijesti poziva se samo metoda za uklanjanje obavijesti, *cancelNotification*. Pritiskom na tipku „Da“ pozivaju se metode *resetNotification*, *changeWallpaper* već spomenuti *cancelNotification*. Prva metoda, *resetNotification*, pruža mogućnost vraćanje originalne pozadinske slike u slučaju da korisnik ne ostaviti sliku koju će aplikacija postaviti. U ovoj metodi se prvo sprema trenutna pozadinska slika kao niz bajtova. Zatim se postavlja obavijest čijim pritiskom se u metodu *onReceive* šalje spremljena pozadinska slika te signal *RESET_ACTION*. Nakon postavljanja navedene obavijesti poziva se metoda *changeWallpaper* (programski kod 6.12) koja za parametre prima prepoznatu emociju i varijablu tipa *boolean* koja govori treba li se postaviti nova pozadinska slika ili vratiti originalna. Pozadinske slike su unaprijed definirane te se nalaze u mapi *assets*. Cijela metoda se sastoji samo od provjere *boolean* varijable te postavljanje pozadinske slike. Dijagram toka postavljanja pozadinske slike se nalazi na slici 6.3.

```
public void changeWallpaper(Context context, String emotion, boolean reset)
{
    wallpaperManager = (wallpaperManager == null) ?
            WallpaperManager.getInstance(context.getApplicationContext()) : wallpaperManager;
    try
    {
        if(reset && wp != null)
        {
            wallpaperManager.setBitmap(wp);
        }
        else if(!reset)
        {
            String wallpaper = "wallpapers/" + emotion + ".jpg";
            InputStream img = context.getAssets().open(wallpaper);
            wallpaperManager.setStream(img);
        }
    } catch (IOException e)
    {
        Log.d("changeWallpaper", e.getMessage());
    }
}
```

Programski kod 6.12. Java kod metode *changeWallpaper* unutar klase *EmotiontHandler*



Slika 6.3. Dijagram toka promjene slike pozadine

Zatim se poziva metoda *cancelNotification* koja uklanja obavijest s upitom za promjenu pozadinske slike prema njegovo identifikacijskom broju, prema kodu:

```
private void cancelNotification(Context context, int id)
{
    NotificationManager notificationManager =
(NotificationManager)context.getSystemService(Context.NOTIFICATION_SERVICE);
    notificationManager.cancel(id);
}
```

Programski kod 6.13. Java kod metode *cancelNotification* unutar klase *EmotiontHandler*

U trećem slučaju u metodi *onReceive* kada je ulazni signal *RESET_ACTION* dohvaća se originalna pozadinska slika te poziva metoda *changeWallpaper* sa zadanim istinitim parametrom tipa *boolean* kako bi se postavila originalna slika, a zatim se poziva već opisani *cancelNotification*.

```
@Override
public void onReceive(Context context, Intent intent)
{
    String action = intent.getAction();
    switch(action)
    {
```

```
        case RESET_ACTION:
            try
            {
                Bundle extras = intent.getExtras();
                byte[] b = extras.getByteArray("picture");

                wp = BitmapFactory.decodeByteArray(b, 0, b.length);
            }
            catch(Exception e)
            {
                Log.d("onReceive", e.getMessage());
            }
            changeWallpaper(context, null, 0, true);
            cancelNotification(context, questionNotificationId);
            cancelNotification(context, resetNotificationId);
            Toast.makeText(context,"reset",Toast.LENGTH_SHORT).show();
            break;
        case YES_ACTION:
            resetNotification(context);
            changeWallpaper(context, getEMOTION(), 1, false);
            cancelNotification(context, questionNotificationId);
            break;
        case NO_ACTION:
            cancelNotification(context, questionNotificationId);
            cancelNotification(context, resetNotificationId);
            break;

        case TEST_YES_ACTION:
            //Toast.makeText(context,"testyes",Toast.LENGTH_SHORT).show();
            new sendPost().execute(getEMOTION(),"1");
            cancelNotification(context, testNotificationId);
            break;
        case TEST_NO_ACTION:
            //Toast.makeText(context,"testno",Toast.LENGTH_SHORT).show();
            new sendPost().execute(getEMOTION(),"2");
            cancelNotification(context, testNotificationId);
            break;
    }
}
```

Programski kod 6.14. Java kod metode *cancelNotification* unutar klase *EmotiontHandler*

Pozivom zadnje metode u metodi *onPostExecute* asinkrone klase *getPostEmotion – quoteNotification*, pomoću klase *DBHelper* iz baze podataka dohvaća se inspirativni citat prigodan za prepoznatu emociju te ispisuje korisniku u obliku obavijesti. Za prikupljanje citata koristila se web stranica izrađena u PHP-u. Ona se sastoji od tekstnog okvira za upis citata, izbornih gumbova (engl. *radio button*) za odabir povezane emocije, tipke za slanje upisanih podataka te na kraju ispis svih unosa u bazu podataka. Pritiskom na tipku podaci se, HTTP POST zahtjevom, šalju u istu PHP skriptu koja prepozna zahtjev te sprema u SQLite bazu podataka. PHP kod skripte nalazi se u P.6.3. Baza podataka sastoji se od dvije tablice: *Emotions* i *Quotes*. U tablici *Emotions* spremljene su sve emocije s kojima aplikacija Emodapter rukuje, a u tablici *Quotes* nalaze se citati povezani s emocijama pomoću stranog ključa. SQL kod za kreiranje i popunjavanje tablice *Emotions* nalazi se u programskom kodu 6.15, a kod za kreiranje tablice *Quotes* u programskom kodu 6.16.

```
CREATE TABLE 'Emotions' (
    'id' INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
    'emotion' TEXT NOT NULL)
INSERT INTO "Emotions" ("emotion") VALUES ('anger');
INSERT INTO "Emotions" ("emotion") VALUES ('disgust');
INSERT INTO "Emotions" ("emotion") VALUES ('fear');
INSERT INTO "Emotions" ("emotion") VALUES ('joy');
INSERT INTO "Emotions" ("emotion") VALUES ('sadness');
```

Programski kod 6.15. SQL kod za izradu i popunjavanje tablice *Emotions*

```
CREATE TABLE 'Quotes' (
    'id' INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
    'quote' TEXT NOT NULL,
    'emotion_id' INTEGER NOT NULL,
FOREIGN KEY (emotion_id) REFERENCES Emotions(id));
```

Programski kod 6.16. SQL kod za izradu tablice *Quotes*

Kako je opisano u poglavlju 5.2, baza podataka se zapisuje u običnu datoteku, u ovom slučaju *Emodapter.db*, koja se nakon završetka upisivanja podataka premješta u mapu *assets* unutar projekta aplikacije Emodapter. Kako bi se baza podataka mogla koristiti u aplikaciji ona se mora prekopirati u lokalnu mapu aplikacije koja je najčešće oblika */data/data/ime.paketa/databases*, tj. u ovom slučaju put mape treba biti: */data/data/com.etfos.kboni.emodapter/databases/Emodapter.db*. Kopiranje iz mape *assets* u lokalnu mapu aplikacije izvršava se pomoću klase *DBHelper* koja je proširena klasom *SQLiteOpenHelper*.

Prvo se poziva metoda *createDataBase* (programski kod 6.17) koja poziva metodu *checkDataBase* (programski kod 6.18) koja provjerava postoji li baza podatak u lokalnoj mapi. Ako ne postoji, onda se poziva metoda *copyDataBase* (programski kod 6.18) koja kopira bazu iz mape *assets* u lokalnu mapu. Nakon toga poziva se metoda *openDataBase* (programski kod 6.19) koja postavlja rukovatelja vezom između baze podataka i aplikacije s mogućnosti čitanja i pisanja.

```
public void createDataBase() throws IOException {
    boolean dbExist = checkDataBase();
    if (dbExist) {
        openDataBase();
    } else {
        this.getReadableDatabase();
        try {
            copyDataBase();
            openDataBase();
        } catch (IOException e) {
            Log.e("createDataBase", e.getMessage());
        }
    }
}
```

Programski kod 6.17. Java kod metode *checkDataBase* unutar klase *DBHelper*

```
private boolean checkDataBase() {

    boolean checkDb = false;
    try {
        String myPath = DB_PATH + DB_NAME;
        File dbfile = new File(myPath);

        checkDb = dbfile.exists();
    } catch (SQLiteException e) {
        Log.e("tle99 - check", e.getMessage());
    }
    return checkDb;
}
```

Programski kod 6.18. Java kod metode *createDataBase* unutar klase *DBHelper*

```
public void copyDataBase() throws IOException
{
    try {
        InputStream myInput = context.getAssets().open(DB_NAME);
        String outputFileName = DB_PATH + DB_NAME;

        OutputStream myOutput = new FileOutputStream(outputFileName);

        byte[] buffer = new byte[1024];
        int length;

        while((length = myInput.read(buffer))>0){
            myOutput.write(buffer, 0, length);
        }

        myOutput.flush();
        myOutput.close();
        myInput.close();
    } catch (Exception e) {
        Log.e("tle99 - copyDatabase", e.getMessage());
    }
}
```

Programski kod 6.19. Java kod metode *copyDataBase* unutar klase *DBHelper*

```
public void openDataBase() throws SQLException
{
    String myPath = DB_PATH + DB_NAME;
    myDB = SQLiteDatabase.openDatabase(myPath, null, SQLiteDatabase.OPEN_READWRITE);
}
```

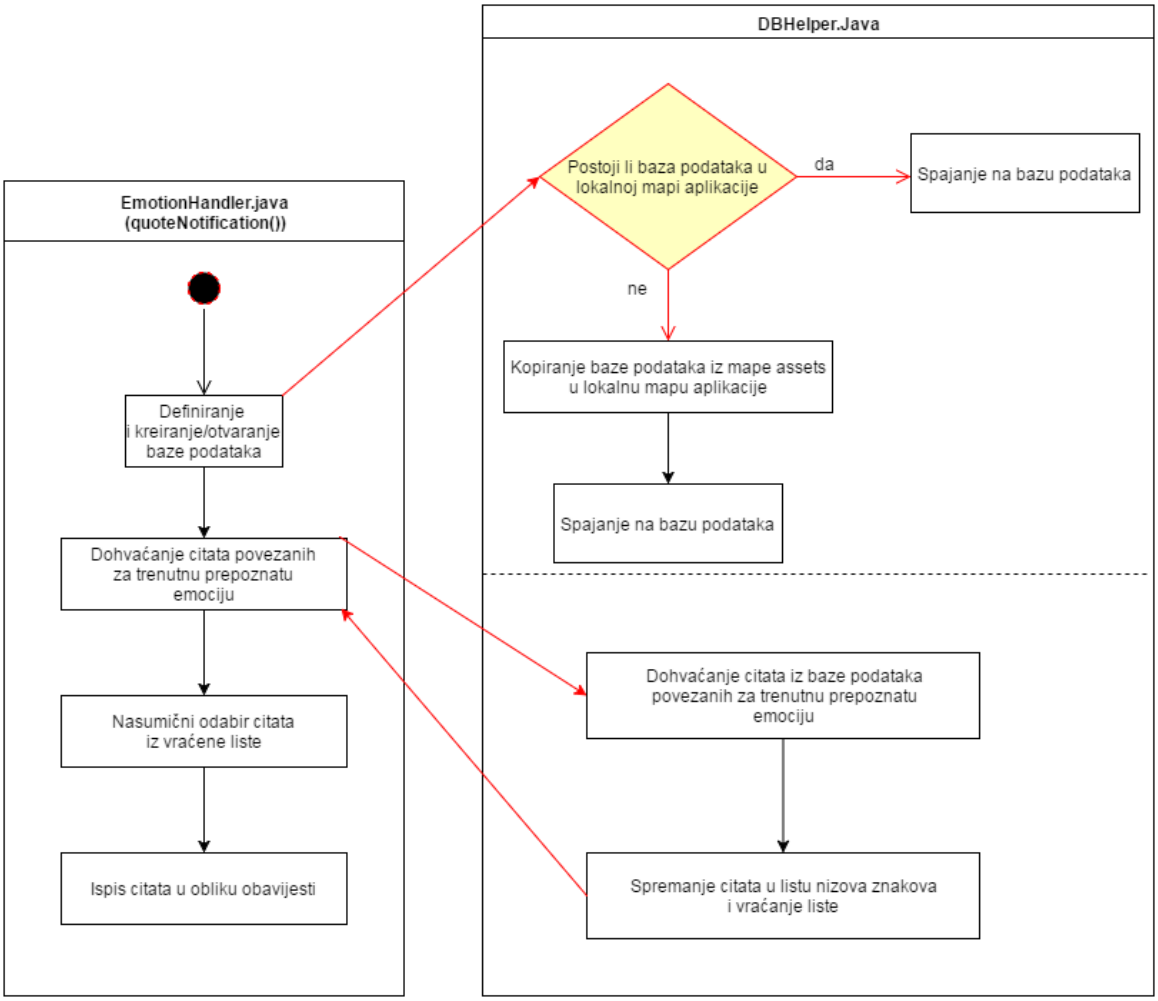
Programski kod 6.20. Java kod metode *openDataBase* unutar klase *DBHelper*

Zatim, za dohvaćanje citata potrebno je pozvati metodu *getQuotes* (programski kod 6.21) koja šalje upit na bazu podataka te vraća listu svih upisanih citata povezanih s prepoznatom emocijom. Dijagram toka ispisa citata je vidljiv na slici 6.4, a kod metode *getQuotes* je sljedeći:

```
public List<String> getQuotes(String emotion)
{
    List<String> listQuotes = new ArrayList<String>();
    SQLiteDatabase db = this.getWritableDatabase();
    Cursor c;

    String sql = "SELECT * FROM " + TB_QUOTES
            + " JOIN " + TB_EMOTIONS + " ON " + TB_QUOTES + ".emotion_id = " + TB_EMOTIONS + ".id"
```

```
            + " WHERE emotion = ?";
    String[] selectionArgs = { emotion };
    try {

        c = db.rawQuery(sql, selectionArgs);
        if(c == null) return null;

        String name;
        c.moveToFirst();
        do {
            name = c.getString(1);
            listQuotes.add(name);
        } while (c.moveToNext());
        c.close();
    } catch (Exception e) {
        Log.e("tle99", e.getMessage());
    }

    db.close();

    return listQuotes;
}
```

Programski kod 6.21. Java kod metode *getQuotes* unutar klase *DBHelper*



Slika 6.4. Dijagram toka ispisa citata u obliku obavijesti

7. NAČIN RADA I TEST APLIKACIJE EMODAPTER

7.1. Aplikacija Emodapter

Veći dio aplikacije je izrađena u okolini Android Studio te je namijenjena za najnoviju Android OS inačicu 6.0, a minimalna podržana inačica s kojom se aplikacija može koristiti je 4.0.3. Nakon instalacije aplikacije potrebno je u postavkama uređaja aktivirati Emodapter kao virtualnu tipkovnicu. Zatim, pri prvom otvaranju virtualne tipkovnice potrebno je postaviti Emodapter kao željenu virtualnu tipkovnicu, a to se postiže spuštanjem statusne trake te pritiskom na obavijest za odabir virtualne tipkovnice. Prvotni izgled virtualne tipkovnice aplikacije Emodapter prikazan je na slici 7.1.

Slika 7.1. Prikaz osnovnog izgleda virtualne tipkovnice pri prvom pokretanju

7.2. Test aplikacije Emodapter

Za testiranje sustava izrađena je još jedna baza podataka, također i PHP skripta te dodatne dvije Java metode u Android aplikaciji. Dvije nove Java metode su: *testNotification* i *sendTestPost* te se one pozivaju iz metode *onPostExecute* unutar klase *getPostEmotion*, opisane u poglavlju 6 (programski kod 6.7). Metoda *testNotification* kreira dodatnu obavijest s upitom prema korisniku je li prepoznata emocija ispravna, prema kodu:

```
public void testNotification(Context c, String emotion)
{
    String question = "Je li Vaša trenutna emocija: " + emotion + "?";
    Intent yes_intent = new Intent(c, EmotionHandler.class);
    yes_intent.setAction(TEST_YES_ACTION);

    Intent no_intent = new Intent(c, EmotionHandler.class);
    no_intent.setAction(TEST_NO_ACTION);

    PendingIntent yes_pIntent = PendingIntent.getBroadcast(c, (int) System.currentTimeMillis(),
yes_intent, PendingIntent.FLAG_UPDATE_CURRENT);
```

```
    PendingIntent no_pIntent = PendingIntent.getBroadcast(c, (int) System.currentTimeMillis(),
no_intent, PendingIntent.FLAG_UPDATE_CURRENT);

    NotificationCompat.Builder mBuilder =
            new NotificationCompat.Builder(c)
                    .setSmallIcon(notificationIcon)
                    .setContentTitle("Emocija: " + emotion)
                    .setStyle(new NotificationCompat.BigTextStyle().bigText(question))
                    .addAction(R.drawable.ok, "Da", yes_pIntent)
                    .addAction(R.drawable.cancel, "Ne", no_pIntent)
                    .setContentIntent(PendingIntent.getActivity(c, 0, new Intent(), 0))
                    .setAutoCancel(true)
                    .setWhen(0).setPriority(Notification.PRIORITY_MAX);


    NotificationManager mNotificationManager =
            (NotificationManager) c.getSystemService(Context.NOTIFICATION_SERVICE);

    mNotificationManager.notify(EmotionHandler.testNotificationId , mBuilder.build());
}
```

Programski kod 7.1. Java kod metode *testNotification* unutar klase *EmotionHandler*

Kao što je vidljivo, u programskom kodu 7.1, dodana su i dva nova signala: *TEST_YES_ACTION* i *TEST_NO_ACTION*, koji se šalju prema *onReceive* metodi klase *EmotionHandler*, koja je opisana u poglavlju 6 (programski kod 6.14). U oba slučaja signala poziva se asinkrona klasa *sendTestPost* koja slično radi kao klasa *getPostEmotion*. Razlika je u tome što se u metodi *doInBackground* klase *sendTestPost* šalje, unaprijed zadana, identifikacija korisnika u obliku niza znakova, zatim, prepoznata emocija te odziv korisnika je li emocija točno prepoznata. Klasa *sendTestPost* se poziva još jednom iz metode *onPostExecute* klase *getPostEmotion* kako bi se prikupile i sve prepoznate emocije korisnika. Završeni kod klase *sendTestPost* prikazuje programski kod 7.2.

```
public class sendTestPost extends AsyncTask<String, String, String>
{
    @Override
    protected String doInBackground(String... args) {

        if(args[0].isEmpty())
        {
            return "";
        }
        String resp = "";
        try
        {
            String emotion = args[0];
            URL url = new URL("http://kboni.site40.net/diplomski/Test.php");
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();

            conn.setReadTimeout(10000);
            conn.setConnectTimeout(15000);
            conn.setRequestMethod("POST");
            conn.setDoInput(true);
            conn.setDoOutput(true);

            Uri.Builder builder = new Uri.Builder();
```

```
            builder.appendQueryParameter("uid", TEST_UID)
                    .appendQueryParameter("emotion", emotion);
            if(!args[1].isEmpty())
            {
                String rec_correct = args[1];
                builder.appendQueryParameter("rec_correct", rec_correct);
            }
            String query = builder.build().getEncodedQuery();
            OutputStream os = conn.getOutputStream();
            BufferedWriter writer = new BufferedWriter(
                    new OutputStreamWriter(os, "UTF-8"));
            writer.write(query);
            writer.flush();
            writer.close();
            os.close();
            int responseCode=conn.getResponseCode();
            if (responseCode == HttpURLConnection.HTTP_OK) {
                String line;
                BufferedReader br=new BufferedReader(new InputStreamReader(conn.getInputStream()));

                while((line=br.readLine()) != null)
                {
                    resp+=line;
                }
            }
        }catch( Exception e)
        {
            Log.v("Async Error", e.getMessage());
        }
        return resp;
    }
    @Override
    protected void onPostExecute(String emotion) {
        Log.v("Async... ", emotion);
    }
}
```

Programski kod 7.2. Java kod klase *sendTestPost* unutar klase *EmotionHandler*

U PHP skripti provjerava se postoje li ulazni podaci (identifikacija i emocija) kao HTTP POST zahtjev te ako postoji odziv korisnika (varijabla *rec_correct*) podaci se spremaju u tablicu *Requests*, a u suprotnom u tablicu *AllEmotion*. Kod PHP skripte se nalazi u P.6.4. Navedene tablice nalaze se u SQLite bazi podataka *Test.db* uz još jednu tablicu: *Emotions*, koja je identična istoimenoj tablici opisanoj u poglavlju 6 (programski kod 6.15). Tablica *Requests* sadrži identifikaciju, strani ključ emocije i odziv korisnika, prema programskom kodu 7.3.

```
CREATE TABLE Requests(
id INTEGER PRIMARY KEY AUTOINCREMENT,
uid CHAR(10),
rec_correct INTEGER,
emotion_id INTEGER NOT NULL,
FOREIGN KEY (emotion_id) REFERENCES Emotions(id));
```

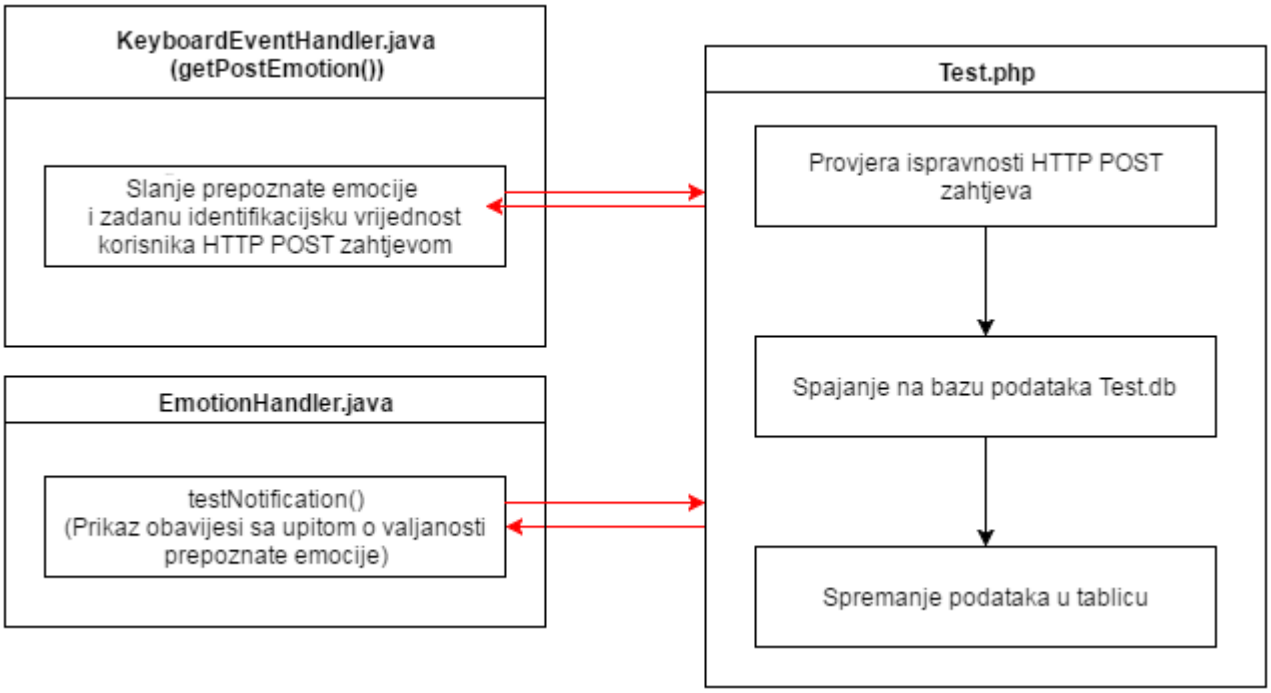
Programski kod 7.3. SQL kod za kreiranje tablice *Requests*

Tablica *AllEmotions* je slična kao tablica *Requests*, s razlikom da ne sadrži redak odziva korisnika *rec_correct*, prema kodu 7.4.

```
CREATE TABLE AllEmotion(
id INTEGER PRIMARY KEY AUTOINCREMENT,
uid CHAR(10),
emotion_id INTEGER NOT NULL,
FOREIGN KEY (emotion_id) REFERENCES Emotions(id));
```

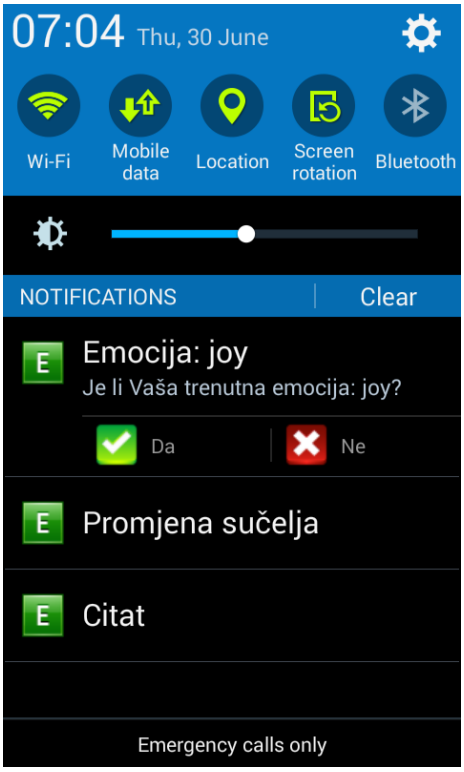
Programski kod 7.4. SQL kod za kreiranje tablice *AllEmotions*

Dijagram toka za testiranje nalazi se na slici 7.2., a izgled obavijesti s upitom na slici 7.3.



Slika 7.2. Dijagram toka skupljanja testnih podataka

Testiranje se izvršava normalnim korištenjem virtualne tipkovnice, odnosno dopisivanjem s drugim ljudima preko bilo koje aplikacije za razmjenu poruka, kao npr. „Messenger“. Kada korisnik završi s pisanjem teksta i „zatvori“ virtualnu tipkovnicu počinje proces prepoznavanja emocije kako je opisano u poglavlju 6. Nakon toga, ispisuju se obavijesti s citatom i s upitom o promjeni pozadinske slike, te dodatna obavijest za testiranje – upit korisnika je li emocija točno prepoznata. Izgled testne obavijesti nalazi se na slici 7.3.



Slika 7.3. Izgled testne obavijesti

Kako je opisano u potpoglavlju 5.3.2., točnost Tone Analyzer API-a se kreće između 41% i 68%, a budući da aplikacija „Emodapter“ koristi i Translate API za prevođenje teksta očekivana točnost prepoznavanja emocije aplikacije je između 30% i 50%.

7.3. Rezultati testiranja

U testiranju je sudjelovalo osam osoba koje su pet dana koristile aplikaciju.. Iz prikupljenih podataka kreirane su tri tablice. Tablica 7.1 pokazuje ukupne postotke prepoznatih pojedinačnih emocija svih korisnika za vrijeme korištenja aplikacije. Tablica 7.2 pokazuje postotke točno, odnosno, netočno prepoznatih pojedinačnih emocija za sve korisnike za vrijeme korištenja aplikacije. U trećoj tablici (tablica 7.3) nalaze se ukupni postoci točno, odnosno, netočno prepoznatih pojedinačnih emocija za sve korisnike za vrijeme korištenja aplikacije.

Tablica 7.1. Ukupni postoci svih prepoznatih emocija

Emocija	Ukupan postotak
ljutnja	10,83%
gađenje	5,42%
strah	17,08%
radost	55,00%
tuga	11,67%

Tablica 7.2. Ukupni postoci emocija za koje su korisnici označili je li prepoznata emocija točna

Emocija	Točni	Netočni
ljutnja	31,58%	68,42%
gađenje	12,50%	87,50%
strah	16,67%	83,33%
radost	53,62%	46,38%
tuga	31,25%	68,75%

Tablica 7.3. Ukupni postoci točnih i netočnih prepoznavanja emocija

Točni	Netočni
38,97%	61,03%

Prema rezultatima u tablici 7.2, vidljivo je da je najveća točnost postignuta za emociju radost, a najnetočnija za emociju gađenje. Međutim, prema tablici 7.3, ukupni postoci zadovoljavaju očekivane rezultate s 38,97% točnosti. Ispitivanjem korisnika, tri od njih osam, odnosno 37,5% je potvrdilo da im je aplikacija uspješno „popravilo“ raspoloženje, tj. emotivno stanje što znači da aplikacija upotpunjava svoj cilj, ali su potrebna daljnja istraživanja i razvoj.

8. ZAKLJUČAK

Cilj ovog diplomskog rada je izrada Android aplikacije kojim se pokušava pozitivno utjecati emotivno stanje korisnika za vrijeme dopisivanja s drugim ljudima preko mobilnog uređaja. Unatoč mnoštvu istraživanja, sustavi za automatsko prepoznavanje emocija iz napisanog teksta nemaju zadovoljavajuće rezultate prepoznavanja. Osim toga, ne postoje potpuno univerzalne metode utjecaja na emocije s elementima koje nudi sučelje mobilnog uređaja, jer svaka osoba je jedinstvena te se najvećim dijelom svaki aspekt, kao na primjer boje i slike, doživljavaju subjektivno. Još jedna velika prepreka je i mnoštvo kultura i jezika što dodatno otežava izradu univerzalnog sustava za prepoznavanje emocija iz napisanog teksta.

Unatoč navedenim problemima, ukupna očekivana točnost je postignuta, ali nažalost taj postotak nije dovoljan za svakodnevnu upotrebu aplikacije „Emodapter“. Međutim, ona se može iskoristiti u svrhe istraživanja na raznim područjima psihologije, ali i marketinga.

U daljnjem razvoju aplikacije trebali bi se koristiti precizniji sustavi za prijevod i prepoznavanje emocija koji u ovom radu nisu korišteni zbog visoke cijene. Nadalje, za točnije prepoznavanje emocija trebali bi se koristiti i drugi oblici informacija osim teksta, kao na primjer brzina tipkanja, učestalost pogrešaka u tipkanju i drugo. Za poboljšanje dijela emotivnog stanja korisnika trebalo bi nastaviti istraživanje na području psihologije te dizajna.

LITERATURA

[1] Social Networking Fact Sheet, Pew Research Center - Internet, Science & Tech, http://www.pewinternet.org/fact-sheets/social-networking-fact-sheet/, 25. lipnja 2015.

[2] Socially Mobile: Does the Smartphone Rule Social Media?, Incite Group, http://www.incite-group.com/customer-engagement/socially-mobile-does-smartphone-rule-social-media, 26. lipnja 2015.

[3] The Psychology of Color and its Meaning, http://www.colorpsychology.org/, 26. lipnja 2015.

[4] J. Grbavac, V. Grbavac, Pojava društvenih mreža kao globalnog komunikacijskog fenomena, Media, culture and public relations, br. 2, sv. 5, str. 206-219, rujan 2014.

[5] K.N. Hampton, L. S. Goulet, K. Purcell, Social Networking Sites and our Lives, How People's Trust, Personal Relationships, and Civil and Political Involvement are Connected to their Use of Social Networking Sites and other Technologies, http://www.pewinternet.org/2011/06/16/social-networking-sites-and-our-lives/ 25. lipnja 2015.

[6] K. Blažeka, Društvene mreže i obrasci komuniciranja mladih, CUC 2010, http://cuc.carnet.hr/2010/ponedjeljak, 25. lipnja 2015.

[7] Is Social Networking Changing Childhood, Common Sence Media, https://www.commonsensemedia.org/about-us/news/press-releases/is-social-networking-changing-childhood, 25. lipnja 2015.

[8] R. Relja, T. Božić, Socio-ekonomski aspekti korištenja mobitela među mladima, Media, Culture and Public Relation, br. 2, sv. 3, str. 138-149, rujan 2012.

[9] A. Damasio, The Feeling of What Happens-Body and Emotion in the making of Consciousness, William Heinemann, London, 1999.

[10] M. Duh, R. Kolar, Basic Emotions and Colours as Perceived by Fourth Grade Pupils, Croatian Journal of Education, br. 3, sv. 16, str. 643-675, listopad 2012.

[11] K. O'Regan, Emotion and E-learning, Journal of Asynchronous Learning Networks, br. 3., sv. 7, str. 78.-92., rujan 2003.

[12] G. Johnson, Theories of Emotion, Internet Encyclopedia of Philosophy - A Peer-Reviewed Academic Resource, http://www.iep.utm.edu/emotion/, 22. lipnja 2015.

[13] Paul Ekman Group LLC, http://www.paulekman.com/paul-ekman/, 30. lipnja 2015.

[14] Color Psychology (the "Colour Affects" system), http://micco.se/wp-content/uploads/2010/05/Micco-Groenholm-on-Color-Affects-System.pdf, 22. lipnja 2015.

[15] A.J. Elliot, M. A. Maier, Color and Psychological Functioning, Current Directions in Psychological Science, br. 5, sv. 16, str. 250-254, lipanj 2007.

[16] K. Sullivan, Perception of Images in Advertising and Impac on Consumer's Lives, Ethica Publishing, http://ethicapublishing.com/ATEOI_ch2.pdf, 23. lipnja 2015.

[17] Four Ways That Inspirational Quotes Can Change Your Life!, Self Growth - The Online Self Improvement Community, http://www.selfgrowth.com/articles/Four_Ways_Inspirational_Quotes_Can_Change_Your_Life.html, 24. lipnja 2015.

[18] S.N. Shivhare, S. Khethawat, Emotion Detection from Text, http://airccj.org/CSCP/vol2/csit2237.pdf, 25. lipnja 2016.

[19] H. Binali, Ch. W., V. Potdar, Computional Approaches for Emotion Detection in Text, 4th IEEE International Conference on Digital Ecosystems and Technologies (DEST 2010), Dubai, United Arab Emirates, 12-15 April 2010 https://www.researchgate.net/publication/224185404_Computational_approaches_for_emotion_detection_in_text, 25.lipjna 2016..

[20] About Android OS, https://developer.android.com/about/index.html, 26. svibnja 2016.

[21] About Android Studio, https://developer.android.com/studio/intro/index.html, 26. svibnja 2016.

[22] About SQLite, https://www.sqlite.org/about.html, 26. svibnja 2016.

[23] Yandex Translate API Documentation https://tech.yandex.com/translate/doc/dg/reference/translate-docpage/, 30. svibnja 2016.

[24] Watson Developer Cloud – Tone Analyzer, API Reference, http://www.ibm.com/watson/developercloud/tone-analyzer/api/v3/, 12.lipnja 2016.

[25] PHP Documentation, http://php.net/docs.php, 12. lipnja 2016.

SAŽETAK

Glavni cilj ovog diplomskog rada bio je izraditi Android aplikaciju, koja pokušava korisnicima poboljšati trenutno emotivno stanje kroz promjene na grafičkom sučelju mobilnog uređaja korisnika, Promjene na grafičkom sučelju uključuju promjenu pozadinske slike i boje tipkovnice te ispisivanje inspirativnih citata u obliku obavijesti. Istraživanje je provedeno na grupi od osam osoba kroz pet dana korištenja aplikacije. Testiranje se sastojalo od korištenja virtualne tipkovnice za vrijeme dopisivanja s osobama. Prema rezultatima istraživanja, aplikacija upotpunjava očekivane kriterije, ali oni su nedovoljni kako bi ona služila za svakodnevnu upotrebu. Unatoč tome, aplikacija se može koristiti za razna istraživanja na raznim područjima psihologije, marketinga, dizajna i tehnologije.

Ključne riječi: analiza teksta, Android, društvene mreže, emocija.

ABSTRACT

Main goal of this master's thesis was to develop an Android application, which is trying to improve users emotional state through changes in the users graphical interface of the mobile device. The changes in the users graphical interface include changing the background picture, the colour of the keyboard and inspirational quotes in the form of push notifications. The application testing was carried out on a group of eight persons during five days period. The testing consisted of using the application's virtual keyboard while messaging with other persons. The results showed the application fulfills the expected testing criteria, but the criteria does not meet the required demands for everyday use. Despite that fact, the application can be used for research in various fields of psychology, marketing, design and technology.

Keywords: textual analysis, Android, social networks, emotion.

ŽIVOTOPIS

Karlo Boni rođen je 25. svibnja 1990. u Osijeku. Osnovnu školu upisao i završio u OŠ „Darda“ u Dardi. Nakon osnovne škole završava prirodoslovno-matematičku gimnaziju u Osijeku. 2008. godine upisuje Sveučilišni preddiplomski studij računarstva na Elektrotehničkom fakultetu u Osijeku te 2011. stječe akademski naziv sveučilišnog prvostupnika inženjera računarstva. Iste godine nastavlja s obrazovanjem na diplomskom studiju Elektrotehničkog fakulteta. Raspolaže aktivnim znanjem engleskog i mađarskog jezika. Uz stečena znanja na fakultetu raspolaže i poznavanjem raznih programskih i skriptnih jezika: C#, HTML, CSS, JavaScript, PHP, Java i Python.

PRILOZI

P.3.1. Psihološka svojstva osnovnih boja

BOJA	Pozitivni utjecaj	Negativni utjecaj
CRVENA	fizička hrabrost, snaga, toplina, energija, nagon za preživljavanjem, "borba ili bijeg", poticanje, maskulinitet, uzbuđenje.	prkos, agresivnost, vizualni utjecaj, pritisak.
PLAVA	inteligencija, komunikacija, povjerenje, učinkovitost, mir, dužnost, logika, hladnoća, odraz, mirnoća.	hladnoća, povučenost, nedostatak emocija, neljubaznost.
ŽUTA	optimizam, samopouzdanje, samopoštovanje, ekstraverzija, emocionalna snaga, prijateljstvo, kreativnost.	iracionalnost, strah, emocionalna krhkost, depresija, anksioznost, samoubojstvo
ZELENA	sklad, balans, osvježenje, univerzalna ljubav, odmor, obnova, ohrabrenje, ekološka svijest, ravnoteža, mir.	dosada, stagnacija, blagost, slabljenje.
LJUBIČASTA	duhovna svijest, ograničavanje, vizija, luksuz, autentičnost, istina, kvaliteta.	introvertnost, dekadencija, potiskivanje, inferiornost.
NARANČASTA	fizička udobnost, hrana, toplina, sigurnost, strast, senzualnost, obilje, zabava.	uskraćivanje, frustracija, neozbiljnost, nezrelost.
RUŽIČASTA	fizički spokoj, odgoj, toplina, ženstvenost, ljubav, seksualnost, opstanak vrste.	inhibicija, emocionalna klaustrofobija, kastracija, fizička slabost.
SIVA	psihološka neutralnost.	nedostatak povjerenja, vlaga, depresija, hibernacija, nedostatak energije.
CRNA	sofisticiranost, glamur, sigurnost, emocionalna sigurnost, učinkovitost, tvar.	ugnjetavanje, hladnoća, prijetnja, težina.
BIJELA	higijena, sterilnost, jasnoća, čistoća, jednostavnost, sofisticiranost, učinkovitost.	sterilnost, hladnoća, prepreke, neljubaznost, elitizam.
SMEĐA	izobilnost, toplina, priroda, pouzdanost, podrška.	nedostatak humora, težina, nedostatak sofisticiranosti.

P.6.1. XML kod položaja, izgleda i dekadskih vrijednosti pojedine tipke tipkovnice

```
<Keyboard xmlns:android="http://schemas.android.com/apk/res/android"
    android:keyWidth="10%p"
    android:horizontalGap="1px"
    android:verticalGap="1px"
    android:keyHeight="40dp"
    >
    <Row>
        <Key android:codes="49" android:keyLabel="1" android:keyEdgeFlags="left"/>
        <Key android:codes="50" android:keyLabel="2"/>
        <Key android:codes="51" android:keyLabel="3"/>
        <Key android:codes="52" android:keyLabel="4"/>
        <Key android:codes="53" android:keyLabel="5"/>
        <Key android:codes="54" android:keyLabel="6"/>
        <Key android:codes="55" android:keyLabel="7"/>
        <Key android:codes="56" android:keyLabel="8"/>
        <Key android:codes="57" android:keyLabel="9"/>
        <Key android:codes="48" android:keyLabel="0" android:keyEdgeFlags="right"/>
    </Row>
    <Row>
        <Key android:codes="113" android:keyLabel="q" android:keyEdgeFlags="left"/>
        <Key android:codes="119" android:keyLabel="w"/>
        <Key android:codes="101" android:keyLabel="e"/>
        <Key android:codes="114" android:keyLabel="r"/>
        <Key android:codes="116" android:keyLabel="t"/>
        <Key android:codes="122" android:keyLabel="z"/>
        <Key android:codes="117" android:keyLabel="u"/>
        <Key android:codes="105" android:keyLabel="i"/>
        <Key android:codes="111" android:keyLabel="o"/>
        <Key android:codes="112" android:keyLabel="p" android:keyEdgeFlags="right"/>
    </Row>
    <Row>
        <Key android:codes="97" android:keyLabel="a" android:keyEdgeFlags="left"/>
        <Key android:codes="115" android:keyLabel="s"/>
        <Key android:codes="100" android:keyLabel="d"/>
        <Key android:codes="102" android:keyLabel="f"/>
        <Key android:codes="103" android:keyLabel="g"/>
        <Key android:codes="104" android:keyLabel="h"/>
        <Key android:codes="106" android:keyLabel="j"/>
        <Key android:codes="107" android:keyLabel="k"/>
        <Key android:codes="108" android:keyLabel="l"/>
        <Key android:codes="40,41" android:keyLabel="( )" android:keyEdgeFlags="right"/>
    </Row>
    <Row>
        <Key android:codes="-1" android:keyLabel="CAPS" android:keyWidth="15%p"
android:keyEdgeFlags="left"/>
        <Key android:codes="121" android:keyLabel="y"/>
        <Key android:codes="120" android:keyLabel="x"/>
        <Key android:codes="99" android:keyLabel="c"/>
        <Key android:codes="118" android:keyLabel="v"/>
        <Key android:codes="98" android:keyLabel="b"/>
        <Key android:codes="110" android:keyLabel="n"/>
        <Key android:codes="109" android:keyLabel="m"/>
        <Key android:codes="46" android:keyLabel="."/>
        <Key android:codes="44" android:keyLabel=","/>
    </Row>
    <Row android:rowEdgeFlags="bottom">
        <Key android:codes="63,33,58" android:keyLabel="\? !" android:keyWidth="15%p"
android:keyEdgeFlags="left"/>
        <Key android:codes="58,59,47" android:keyLabel=": ; /" android:keyWidth="15%p" />
        <Key android:codes="32" android:keyLabel="SPACE" android:keyWidth="30%p"
android:isRepeatable="true"/>
        <Key android:codes="-5" android:keyLabel="DEL" android:keyWidth="20%p"
android:isRepeatable="true"/>
        <Key android:codes="-4" android:keyLabel="DONE" android:keyWidth="20%p"
android:keyEdgeFlags="right"/>
    </Row>
</Keyboard>
```

P.6.2. PHP skripta za dohvaćanje prepoznate emocije

```php
<?php

if ($_SERVER['REQUEST_METHOD'] === 'POST' && !empty($_POST['text']))
{
    echo getEmotion(translate($_POST['text']));
}

function getEmotion($text)
{
    $url = 'https://gateway.watsonplatform.net/tone-
analyzer/api/v3/tone?version=2016-05-19&tones=emotion';
    $username = '6f7a9cff-6d73-4f95-9fca-e3097c97eff2';
    $password = '6KGsmNmsLwmo';
    $post = json_encode(array(
        'text' => $text,
    ));
    $header = array(
        'Content-Type: text/plain',
        'Content-Length: ' . strlen($post)
        );
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, $url);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_USERPWD, "$username:$password");
    curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_ANY);
    curl_setopt($ch, CURLOPT_POST, 1);
    curl_setopt($ch, CURLOPT_POSTFIELDS, $post);
    curl_setopt($ch, CURLOPT_HTTPHEADER, $header);
    $json = curl_exec($ch);
    curl_close($ch);

    $emos = json_decode($json, true);
    $emoval = 0;

    foreach ($emos['document_tone']['tone_categories'][0]['tones'] as $array)
    {
        if($array['score'] > $emoval)
        {
            $emoval = $array['score'];
            $emotion = $array['tone_id'];
        }
    }
    return $emotion;
```

```
}

function translate($text)
{
    $url = 'https://translate.yandex.net/api/v1.5/tr.json/translate';
        $postData = array(
         'key' =>
'trnsl.1.1.20160615T172806Z.956320fc38e8d026.6f7244fe4ab4dd5268c3b7fb8636879ce50
dd952',
         'lang' => 'hr-en',
                'text' => $text);

        $options = array(
            'http' => array(
                'header'  => "Content-type: application/x-www-form-
urlencoded\r\ncharset=utf-8\r\n",
                'method'  => 'POST',
                'content' => http_build_query($postData)
            )
        );
        $context  = stream_context_create($options);
        $result = file_get_contents($url, false, $context);
        if ($result === FALSE)
        {
                die("Failed!");
        }
        else
    {
         $response = json_decode($result,true);
         if($response['code'] == 200)
         {
             return $response['text'][0];
         }
    }
}
?>
```

P.6.3. PHP skripta za prikupljanje inspirativnih citata

```
<!doctype html>
<html>
<meta charset="UTF-8">
<?php
mb_internal_encoding("UTF-8");
$db = new PDO('sqlite:Emodapter.db');
$db->setAttribute(PDO::ATTR_ERRMODE,
                        PDO::ERRMODE_EXCEPTION);

if(!empty($_POST['cite']) && !empty($_POST['emotion'])){
        $db->exec('INSERT INTO Quotes(quote,emotion_id) VALUES
(null,"'.$_POST['cite'].'","'.$_POST['emotion'].'");');
}
echo '<form action = "'.$_SERVER['PHP_SELF'].'" method = "POST">';
echo 'Citat:<br>
        <textarea rows="10" cols="50" name = "cite"></textarea><br>
        Emocije:<br>';

$emotions = array();
$emo = $db->query('SELECT * FROM Emotions');
foreach ($emo as $row) {
    $emotions[$row['id']] = $row['emotion'];
    echo '<input type="radio" name="emotion"
value="'.$row['id'].'">'.$row['emotion'].'<br>';
}
        echo '<input type = "submit"></form><br><br>';

$result = $db->query('SELECT * FROM Quotes ORDER BY emotion_id');
echo '<table border = "1">';
echo '<tr><th>ID</th>
                <th>Emocije</th>
         <th>Citat</th></tr>';
foreach($result as $row){
        print "<tr><td>".$row['id']."</td>";
        print "<td>".$emotions[$row['emotion_id']]."</td>";
        print "<td>".$row['quote']."</td></tr>";
}

echo '</table>';
?>
</html>
```

P.6.4. PHP skripta za prikupljanje testnih podataka

```
<?php
  date_default_timezone_set('UTC');

  try {

    if(!empty($_POST['uid']) && !empty($_POST['emotion']))
    {
        $file_db = new PDO('sqlite:Test.db');
        $file_db->setAttribute(PDO::ATTR_ERRMODE,
                               PDO::ERRMODE_EXCEPTION);

      if(!empty($_POST['rec_correct']))
      {
          $insert = "INSERT INTO Requests (uid, emotion, rec_correct)
                     VALUES (:uid, :emotion, :rec_correct)";
          $stmt = $file_db->prepare($insert);
          $stmt->bindParam(':uid', $_POST['uid']);
          $stmt->bindParam(':emotion', $_POST['emotion']);
          $stmt->bindParam(':rec_correct', $_POST['rec_correct']);
          $stmt->execute();
      }
      else
      {
          $insert = "INSERT INTO AllEmotions (uid, emotion)
                     VALUES (:uid, :emotion)";
          $stmt = $file_db->prepare($insert);
          $stmt->bindParam(':uid', $_POST['uid']);
          $stmt->bindParam(':emotion', $_POST['emotion']);
          $stmt->execute();
      }
    }
  }
  catch(Exception $e) {
    echo $e->getMessage();
  }
?>
```